NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

NORMAL APPROXIMATION FOR RESPONSE TIME IN A PROCESSOR-SHARED COMPUTER SYSTEM MODEL

by

Suriya Pornsuriya

March 1984

Thesis advisor: P. A. Jacobs

Approved for public release; distribution unlimited

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Normal Approximation for Response Time in a Processor-Shared Computer System Model		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis March 1984
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Suriya Pornsuriya		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943		12. REPORT DATE March 1984
		13. NUMBER OF PAGES 107
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Response Time, Central Limit Theorem, Heavy Traffic

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In a time-shared computer system, the processor allocates its processing time equally to all jobs submitted for service from a fixed number of terminals. Under Markov assumptions, i.e., independent identically distributed exponential terminal think times and job requested service times, the distributionof response time of a tagged job theoretically can be determined by solving a system of differential equations derived for each initial system state. However, explicit closed form solutions to these equations are quite

complex. The Central Limit Theorem and heavy traffic arguments suggest normal approximations to the distribution of the response time. Simulation of the response time is used to study the accuracy of these normal approximations to the response time distribution via moments and quantiles. Finally, the analysis is extended to a model for a system with two types of terminals.

Normal Approximation for Response Time in a Processor-Shared Computer System Model

by

Suriya Pornsuriya
Ensign, Royal Thai NAvy

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
March 1984

ABSTRACT

In a time-shared computer system, the processor allocates its processing time equally to all jobs submitted for service from a fixed number of terminals. Under Markov assumptions, i.e. independent identically distributed exponential terminal think times and job requested service times, the distribution of response time of a tagged job theoretically can be determined by solving a system of differential equations derived for each initial system state. However, explicit closed form solutions to these equations are quite complex. The Central Limit Theorem and heavy traffic arguments suggest normal approximations to the distribution of the response time. Simulation of the response time is used to study the accuracy of these normal approximations to the response time distribution via moments and quantiles. Finally, the analysis is extended to a model for a system with two types of terminals.

TABLE OF CONTENTS

I. INTRODUCTION 7
- A. BACKGROUND 7
- B. PROCESSOR-SHARING SYSTEM. 7
- C. MODEL AND APPROXIMATE DISTRIBUTION 9

II. MEAN AND MOMENTS FOR A SINGLE JOB TYPE MODEL . . . 12
- A. INTRODUCTION 12
- B. CONDITION ON REQUIRED TIME AND SYSTEM STATE. 13
- C. CONDITION ON REQUIRED TIME 14
- D. MOMENTS AND VARIANCE OF RESPONSE TIME 16
- E. NUMERICAL RESULTS 18

III. SIMULATION FOR ONE JOB TYPE MODEL 20
- A. INTRODUCTION 20
- B. WORK TIME 21
- C. SIMULATION FOR A 2-TERMINAL SYSTEM 22
  - 1. Algorithm 22
  - 2. Numerical Results 24
- D. SIMULATION FOR AN N-TERMINAL SYSTEM 25
  - 1. Algorithm 25
  - 2. Moments of Response Time 28
  - 3. Computation of Simulated Variance 30
  - 4. Computation of Simulated Skewness 30
  - 5. Computation of Simulated Kurtosis 31
  - 6. Standard Error of the Mean Response Time 32
  - 7. Numerical Results 33

IV. NORMAL APPROXIMATION FOR RESPONSE TIME 36
A. INTRODUCTION 36
B. APPROXIMATION BY CENTRAL LIMIT THEOREM 37
1. A Central Limit Theorem for W(t) 37
2. A Central Limit Theorem for response time, R(T) 39
C. APPROXIMATION BY HEAVY TRAFFIC ANALYSIS 40
D. COMPARISON TO SIMULATION DATA 42

V. MODEL FOR A SYSTEM WITH TWO TYPES OF TERMINALS . . 50
A. INTRODUCTION 50
B. STEADY-STATE DISTRIBUTION 51
C. A CENTRAL LIMIT THEOREM FOR THE RESPONSE TIME 55
D. HEAVY TRAFFIC APPROXIMATION FOR THE RESPONSE TIME 58
E. SIMULATION 63
1. Algorithm 63
2. Computation of Simulated Mean, Variance, Skewness and Kurtosis 65
3. Numerical Results 68

VI. SUMMARY AND CONCLUSION 79
A. SUMMARY 79
B. CONCLUSION 81

APPENDIX A: SIMULATION PROGRAM FOR ONE-TYPE MODEL . . . 83

APPENDIX B: SIMULATION PROGRAM FOR TWO-TYPE MODEL . . . 93

LIST OF REFERENCES 104

INITIAL DISTRIBUTION LIST 106

I. INTRODUCTION

A. BACKGROUND

When a computer user, from his terminal, submits a job to the computer, it would be desirable for him that the job be processed right away. However, this rarely happens in the real world, because a computer system usually consists of only one Central Processing Unit (CPU) and many terminals. So if more than one person uses the computer, there will be jobs that request processing at the same time. Hence, some kind of queueing system or time-sharing technique will be needed to organize the allocation of processing time to those submitted jobs.

Generally, a computer system has two types of processing time allocation. One is called "Batch Processing". All the jobs submitted this way form a kind of queue and wait to be served (processed) according to the well known First-Come-First-Served (FCFS) policy, i.e. the first submitted job is processed to completion, then the computer is set up for the next waiting job; in some systems a priority policy based on the length of a job is used to determine the next job to get dedicated use of the processor. The other one which is of interest for this thesis is called "Time-Sharing Processing". It is based on a technique that permits concurrent processing of two or more jobs. Each job no matter when it is submitted gets an equal share of processing time until completion.

B. PROCESSOR-SHARING SYSTEM.

The so-called "Processor-Sharing" or "pure time-sharing" in computer engineering is the system in which the processor

shares its service (processing time) equally among all jobs submitted. In other words, if an individual job requiring a certain amount of processing time is tagged and submitted to the system and finds (j-1) other jobs being processed, then from now on all j jobs, will each receive service (processing time) equal to (1/j)-th of a (processing) time unit per time unit. Of course the rate at which submitted jobs receive service changes each time a new arrival joins the system and each time a completed job departs.

This abstraction of computer capacity allocation may be described in more formal terms as follows : if the chance that any single job, processed alone, finishes in time interval (t,t+h) is $\mu h + o(h)$, (exponential-Markov service), then the chance that a particular "tagged" job in the company of (j-1) others finishes in (t,t+h) is $\mu(h/j)+o(h)$ as h approaches zero.

Clearly, there is no waiting line in processor sharing of the above type. It permits short jobs access to processing right away even if they arrive after longer jobs.

Processor sharing is an approximation to the processor sharing "Round-Robin" model. In this model once a particular tagged job enters the system, it joins the end of an ordered queue. When it reaches the service point it is allocated a fixed quantum (q) of service time. If the job completes within this time it simply leaves the system. If after q seconds it still requires more service, it is immediately returned to the end of the queue. This process then goes on and on until the required service is completed.

In the limit, however, as q approaches zero, the Round-Robin system becomes the Processor-Sharing system. The latter system here will be studied for its characteristics relating to the distribution of response time, i.e. the time that a tagged job requiring a certain amount of processing time actually taken to complete the service and leave the system.

C. MODEL AND APPROXIMATE DISTRIBUTION

Apparently the first study of delays to arriving jobs under processor sharing was conducted by Coffman, Muntz, and Trotter (1970). **[Ref. 1]** They assumed a steady state M/M/1 system under processor sharing, i.e. Poisson arrivals and exponential service times with a single server processor. They then derived an expression for the Laplace transform of the waiting time distribution of an arriving job conditioned on the processing time it requires and the number of jobs it finds in the system on arrival.

The properties of the response time, R, given the processing time required by the arriving job under processor sharing system was further analyzed by D. Mitra (1981) **[Ref. 2]** for the following model. A system consisting of N terminals and a single computer (CPU) can be modeled as a classical machine-repair situation : each thinking terminal (failure-prone machine) applies for computer service at rate λ , and queued or waiting jobs are served at rate μ as long as any jobs are present. If Markov assumptions are made throughout, then X(t), the number of jobs at the service stage, is a birth and death process with transition rates :

$$
\begin{aligned}
X(t) = j \dashrightarrow X(t+h) &= j+1 \quad : \quad \lambda_j h + o(h) \\
\dashrightarrow X(t+h) &= j-1 \quad : \quad \mu_j h + o(h) \qquad (1.1) \\
\dashrightarrow X(t+h) &= j \qquad\ \ : \quad 1 - (\lambda_j + \mu_j) h + o(h)
\end{aligned}
$$

where $\lambda_j = \lambda(N-j)$ is the rate at which a job is submitted to the computer when there are already j jobs in the system, and $\mu_j = \mu$ for $j \geq 1$, otherwise being zero, is

the rate at which the computer gives service to all j jobs submitted. Based on the above transition rates, the distribution of response time under processor sharing is characterized, and the moments such as mean and variance are found under interesting conditions, i.e. the conditional response time, given only the processing requirement T time units, derived from the condition that the tagged job requiring T time units of processing arrives to find (j-1) others in the system. [Ref. 2]

Gaver, Jacobs and Latouche [Ref. 3] have generalized and extended the previous analysis by introducing the idea of processor sharing in an arbitrary birth and death process environment, thus allowing quite general terminal-computer interactions to be represented. In the process, the meaning of "system state at the moment of tagged job arrival" is also clarified by Lavenberg and Reiser. [Ref. 4] Response time characteristics are computed under the assumptions that processor-sharing service rates are processor-state-dependent in a more general way than that described earlier; this allows for approximate representation of overhead penalties and also of job scheduling. Other characteristics of tagged job response are also studied, e.g. the accumulated processing work, W(t'), actually performed on that job by elapsed time t', with t' < T (T = required processing time) following job introduction; note that W(R) = T, so the first passage of W(t') to T is actually the response time. Although differential equations may be obtained for transforms of W(t') under various initial conditions, and hence, implicitly for its distribution, the results are far from being explicit and informative. However, central limit theorems for additive functionals of Markov processes, or for cumulative processes, allow the conclusion that the accumulated work accomplished by fixed time t' on a "long" job is also approximately normally distributed.

Additionally, a normal approximation is shown to be valid for the simple model --and probably for others as well-- when the number of competing terminals becomes large , i.e. under heavy traffic conditions. The quality of the normal approximations for finite job lengths and for a finite number of terminals will be assessed by simulation methods in chapter IV of this thesis.

The differential equations for the mean and moments of the response time of a tagged job requiring a fixed amount of processing time, given that it enters to find an initial number of other jobs being processed, will be derived in chapter II. To remove the condition of initial system state, we will use the steady-state distribution of the number of jobs at the service stage. This will also be explained in the same chapter. A procedure to simulate the response time, given again an initial system state, will be described in chapter III. The empirical response times obtained from simulation will then be considered as stratified random samples. Hence, a method of computing the central moments will be given accordingly.

In chapter V, we will study a bivariate birth and death process model for a computer system having two types of terminals. This model allows relaxation of the independent identically distributed exponential service requirement and terminal think times of the model described by Mitra. [Ref. 2] Under the same conditions as the previous simple model with one job type, we will derive normal approximations for the distribution of the response time of a tagged job requiring t units of processing time. As before, simulation methods will be used to assess the accuracy of normal approximation to the distribution of the response time.

II. MEAN AND MOMENTS FOR A SINGLE JOB TYPE MODEL

A. INTRODUCTION

In this chapter, we consider the birth-death process model with rates in (1.1). Even though the unconditional mean and variance of response time of a Processor-Sharing system may be obtained by deriving the Laplace transform of the equilibrium waiting time distribution [Ref. 1] it is also interesting to develop some results for the conditional expectation of response time of a job requiring T units of processing time, since in the real world one might rather wonder how long a job that requires certain amounts of processing time will be delayed after being submitted to the system.

We will show that the equilibrium mean waiting time in the Processor-Sharing system varies linearly with the service time requirement T, i.e. $E(X) = \lambda T/(\mu(1-\lambda/\mu))$, as $T \longrightarrow \infty$. Thus for arrivals having a service time requirement less than the average, i.e. $T < 1/\mu$, the mean response time is less in the Processor-Sharing system than in the First-Come-First-Served system.

To derive the conditional mean response time, a given tagged job, i.e. a particular job that enters to find (j-1) others waiting for service time, and that requires "T" units of processing time will be considered. Then under Markov assumptions a system of differential equations will be established to allow computation of the conditional mean response time by numerical methods. Other moments may also be obtained by means of solving other systems of differential equations.

B. CONDITION ON REQUIRED TIME AND SYSTEM STATE.

Under the assumptions of a Markov process, i.e. birth and death process, on the number of jobs at the service stage of a system of "N" terminals and a single computer (CPU) with transition rates as in equation (1.1), a system of differential equations for the mean response time may be derived as follows.

Let R refer to the response time of a newly arrived job, and

$$m_j(T) = E[R \mid X(0) = j, W(R) = T], \qquad (2.1)$$

the conditional expectation of the response time, given that the tagged job is initially in the company of (j-1) others, i.e. arrives to find (j-1) jobs present, and requires "work" or processing time equal to T.

Let λ_j and μ_j be as in (1.1). Consider all the possible system changes in (0,h), and subsequently; any of the following mutually exclusively events may occur:

(a) new job arrival, bringing the state to j+1, an event of probability $\lambda_j h$,

(b) accompanying job departure and return to think mode, an event of probability $(j-1)\mu(r(j)/j)h$,

(c) no change in accompanying system state but a reduction in remaining tagged-job service of $(r(j)/j)h$, an event of probability $1-(\lambda_j+(j-1)\mu(r(j)/j))h$.

All other possible events are of probability o(h) and may be ignored.

The term r(j) used above represents the fraction of time the processor actually spends processing when there are j jobs being processed. The fact that r(j) < 1 represents overhead.

Letting $\tilde{\mu}_j = (j-1)\mu(r(j)/j)$, (a), (b) and (c) lead to :

$$m_j(T) = h + m_j(T-(r(j)/j)h)[1-(\lambda_j + \tilde{\mu}_j)h] \tag{2.2}$$

$$+ \lambda_j h m_{j+1}(T-(r(j)/j)h) + \tilde{\mu}_j h m_{j-1}(T-(r(j)/j)h) + o(h).$$

Subtract $m_j(T-(r(j)/j)h)$ from each side , then divide by h and let h ---> 0 to get the differential equations :

$$(r(j)/j) m_j'(T) = 1 - (\lambda_j + \tilde{\mu}_j)\, m_j(T) \tag{2.3}$$

$$+ \lambda_j m_{j+1}(T) + \tilde{\mu}_j m_{j-1}(T).$$

This is a standard system of linear differential equations , initial conditions are $m_j(0) = 0$ for all j . A solution can be obtained in terms of Laplace transforms, by exponential formulas involving matrices, or, numerically, by use of standard computer codes for the solution of systems of linear differential equations. Simple explicit and comprehensible closed form results do not seem attainable.

C. CONDITION ON REQUIRED TIME

The condition that the tagged job entered to find (j-1) others in the system, i.e. X(0) = j, can be removed according to the stationary distribution that corresponds to the system state found by the arriving job. The resulting expression allows the conclusion that the expected response time is "<u>linear</u>" in the required processing time, T. The result here holds for quite general birth-and-death process model, not only for the simple machine-repair setup detailed here. [Ref. 5]

The derivation of linearity of the expected response time is developed, in outline, as follows.

First, observe that the long-run distribution of X(0), i.e. the number of jobs present (including the tagged job) just after the tagged job enters, is

$$q_j = c\pi_{j-1}\lambda_{j-1} = c\pi_j\mu r(j) \quad , \; j = 1,2,\dots,N \qquad (2.4)$$

where c is selected so that the q 's sum to one, and for all j's, $\pi_j = \pi_0(\lambda_0\lambda_1\cdots\lambda_{j-1})/(\mu_1\mu_2\cdots\mu_j)$ is the stationary distribution (assumed to exist) of the Markov chain X(t), i.e. the number of jobs at the service stage, with rates as in (1.1) with $\mu_j = \mu\, r(j)$. The equation (2.4) is intuitively apparent, for the long-run probability that a transition from j-1 to j occurs in (t,t+h) is $\pi_{j-1}\lambda_{j-1}h$ as h ---> 0 and hence equation (2.4) follows by normalization. A formal proof can be provided based either upon an embedded Markov chain formulation, or upon the theory of additive functionals of a Markov process. [Ref. 6] The distribution $\{q_j\}$ has also been given by Kelly. [Ref. 7]

Next, use equation (2.4) to remove the condition that X(0) = j. Put

$$m(T) = E_{X(0)}E[R \mid X(0), W(R)=T] = \sum_{j=1}^{N} q_j m_j(T) . \qquad (2.5)$$

Then in terms of the differential equations (2.3); after multiplying through by j/r(j), one obtains, with initial conditions $m_j(0) = 0$,

$$m'(T) = \sum_{j=1}^{N}(j/r(j))q_j + \sum_{j=1}^{N} q_j (j/r(j))[-(\lambda_j+\tilde{\mu}_j)m_j(T)$$
$$+ \lambda_j m_{j+1}(T) + \tilde{\mu}_j m_{j-1}(T)]$$

$$m'(T) = \sum_{j=1}^{N} (j/r(j))\, q_j . \qquad (2.6)$$

Thus it follows that the long-run conditional expected response time of the processor-sharing system is linear in the processing time requirement T:

$$E[R \mid W(R)=T] = T \sum_{j=1}^{N} (j/r(j))\, q_j$$

$$= TE[X(0)/r(X(0))]. \qquad (2.7)$$

Apparently no such simple form exists for $\mathrm{Var}[R \mid W(R)=T]$, although Mitra [Ref. 2] has given a formula for a particular case. It will be shown, however, that the above variance is indeed proportional to T if T is large.

D. MOMENTS AND VARIANCE OF RESPONSE TIME

The conditional moments for response time of a job requiring T units of processing time may be obtained by a similar derivation to that used for the expected conditional response time. For example, to find an expression in differential equation form of the second moment, one has to consider all the possible system changes during time period (0,h) as has been done for the mean response time in previous section. If the conditional second moment of response time of a tagged job that requires T units of processing time is:

$$m^2_j(T) = E[R^2 \mid X(0)=j, W(R)=T], \qquad (2.8)$$

then, the following results subsequently occur:

$$m^2_j(T)-m^2_j(T-(r(j)/j)h) \tag{2.9}$$

$$= 2m_j(T-(r(j)/j)h)-(\lambda_j+\tilde{\mu}_j)m^2_j(T-(r(j)/j)h)$$

$$+\lambda_j m^2_{j+1}(T-(r(j)/j)h) + \tilde{\mu}_j m^2_{j-1}(T-(r(j)/j)h).$$

As h ---> 0 one obtains the differential equations:

$$(r(j)/j)[dm^2_j(T)/dT] = 2m_j(T)-(\lambda_j+\tilde{\mu}_j)m^2_j(T)$$

$$+\lambda_j m^2_{j+1}(T)+\tilde{\mu}_j m^2_{j-1}(T). \tag{2.10}$$

Again, the distribution found in (2.4) can be used to remove the condition that the job enters to find j-1 others in the system.

The variance may then be computed by usual formula, i.e.

$$\mathrm{Var}[R(T)] = E[R^2(T)] - [E[R(T)]]^2.$$

Likewise, the differential equations for the third and fourth moments, $m^3_j(T)$ and $m^4_j(T)$, may be obtained by the procedure used to evaluate the first and second moments. The expressions for these moments are as follow:

$$(r(j)/j)[dm^3_j(T)/dT] = 3m^2_j(T)-(\lambda_j+\tilde{\mu}_j)m^3_j(T)$$

$$+\lambda_j m^3_{j+1}(T)+\tilde{\mu}_j m^3_{j-1}(T), \tag{2.11}$$

$$(r(j)/j)[dm^4_j(T)/dT] = 4m^3_j(T)-(\lambda_j+\tilde{\mu}_j)m^4_j(T)$$

$$+\lambda_j m^4_{j+1}(T)+\tilde{\mu}_j m^4_{j-1}(T). \tag{2.12}$$

Once the condition of initial system state is removed using the expression of (2.4) we can compute the third and fourth central moments by expanding the powers in order to calculate them in terms of moments around the origin obtained from solving the system of differential equations described above. The central moments may be expressed in the following forms:

$$E[(R(T)-E[R(T)])^3] = E[R^3(T)]-3E[R^2(T)]E[R(T)] + 2[E[R(T)]]^3$$

$$E[(R(T)-E[R(T)])^4] = E[R^4(T)]-4E[R^3(T)]E[R(T)] + 6E[R^2(T)][E[R(T)]]^2-3[E[R(T)]]^4.$$

The skewness and kurtosis of response time are then computed as follows:

$$\text{Skewness} = E[(R(T)-E[R(T)])^3]/(\mathrm{Var}[R(T)])^{3/2},$$

$$\text{Kurtosis} = \{E[(R(T)-E[R(T)])^4]/((\mathrm{Var}[R(T)])^2)\}-3.$$

E. NUMERICAL RESULTS

The conditional expected response time $m_j(T)$ and the conditional moments $m^2_j(T)$, $m^3_j(T)$ and $m^4_j(T)$ can be computed by solving the differential equations (2.10)-(2.12) using numerical methods, i.e. either linear or Runge-Kutta methods. Once these results are obtained, the condition that the tagged job enters to find j-1 others in system is removed. Hence we will obtain the mean and the second, third and fourth moments of response time of a job that requires T units of processing time. These values can then be used to compute the central moments, and eventually allow us to determine the variance, skewness and kurtosis for the distribution of response time.

Table I shows the means, variances, skewnesses and kurtosises of response time of a job requiring T units of processing time for a system of 2 terminals with arrival rate $\lambda = 1$ and service rate $\mu = 2$. We see that as the required work time becomes large the distribution of the response time is moderately close to a symmetric one, since the skewnesses are small. The kurtosis is also approaching zero as the required work time becomes large. This suggests that the distribution of the response time may be approximated by a normal distribution for large required work time.

TABLE I

Numerical Results for 2 Terminals

$\lambda = 1, \quad \mu = 2$

Time	Mean	Variance	Skewness	Kurtosis
0.1000	0.1335	0.0020	0.6633	-1.3415
0.2000	0.2668	0.0074	0.7078	-1.3042
0.3000	0.4002	0.0152	0.7053	-1.2428
0.5000	0.6668	0.0358	0.6876	-1.0823
1.0000	1.3325	0.1031	0.5729	-0.6024
2.0000	2.6719	0.2656	0.5118	-0.4626
3.0000	4.0084	0.4569	0.4614	-0.3802
5.0000	6.6815	0.7711	0.2554	-0.1976

III. SIMULATION FOR ONE JOB TYPE MODEL

A. INTRODUCTION

The numerical method of computing the moments of the conditional response time of a tagged job that requires some fixed amount of processing time indicated in the previous chapter is generally sufficiently accurate, especially if carried out by the Runge-Kutta method. However, the distribution of the response time is also of interest. We will use simulation to study this distribution. We will describe a simulation routine for response time of a job requiring a fixed amount of processing time for the model with one job type described in chapters I and II.

The conditional response time of a tagged job that enters to find j-1 others initially present in the system, and requires T units of processing time can be simulated for the model if the job submission rate to the processor of each terminal and the processing rate of the processor, i.e. λ and μ, are known. Under Markov assumptions, the number of jobs, including the tagged one, in the Processor-Sharing system which consists of one processor and N terminals is considered as a birth and death process with transition rates $\lambda_j = \lambda(N-j)$ and $\mu_j = \mu(j-1)r(j)/j$, where $r(j)$ is defined as a fraction of time the processor actually spends processing when there are j jobs being processed including the tagged job. Thus the interarrival time and the departure time (work completed) of the jobs in the system are exponentially distributed with parameters λ_j and μ_j respectively. We use the LLRANDOMII package available for the Naval Postgraduate School computer system to generate the two exponential times with rates λ_j and μ_j respectively.

The generated arrival and departure times are compared and the sojourn time in state j is determined as well as the next state of the system and the amount of processing time the tagged job gets from the processor during the sojourn time. We then repeat the procedure until the accumulated processing time for the tagged job meets the requirement of work required to completion.

B. WORK TIME

It turns out to be especially convenient to measure time in terms of the amount of actual work or processing that has been accomplished on the tagged job. Let $C(w)$ denote the number of jobs undergoing service at a moment when exactly w units of processing have been accomplished on the tagged job. We will also assume $r(j) = 1$ for all j. The rate of accretion of clock or response time at work time w is $C(w)$: if $C(w) = 1$ then the tagged job is alone and response (clock) time and work time advance at the same rate, while if $C(w) = 17$ the tagged job is accompanied by 16 others and 17 units of response time accrue for every single work time unit. It follows that the response time for the tagged job requiring T units of processing time is simply

$$R(T) = \int_0^T C(w)\,dw.$$

The process $\{C(w)\}$ is a birth and death process related to $X(w)$. It has arrival and departure rates $\lambda_j^w = \lambda j(N-j)$ and $\mu_j^w = \mu j$. All the simulations described in this thesis will be done in work time.

To obtain the conditional expected response time of a tagged job that requires T units of work time as in previous chapter, we remove the condition that the tagged job entered to find j-1 others initially present in the system by applying the same steady-state distribution of the number of jobs in system, q , found in (2.4).

C. SIMULATION FOR A 2-TERMINAL SYSTEM

1. Algorithm

We will first describe the simulation by considering a simple computer system which consists of one processor and only two terminals. Each terminal submits jobs to be processed by the processor at rate λ and the processor has a service rate of μ for jobs already present in the system. The service effort is allocated equally to all jobs present in the system at any time. Therefore, if a job requiring T units of work time enters to find the system empty, the arrival rate of the other job will be λ while there will be no departure. Similarly, if the job enters to find the other one already present in the system, there will be no arrival while the service rate will be $\mu r(2)/2$.

To transform the rates into the terms of work time we multiply them by $1/r(1)$ and $2/r(2)$ respectively. Thus, the arrival rate, if any, becomes $\lambda/r(1)$ and the service rate, if any, is simply μ.

Based on the above transition rates an algorithm to perform simulation for the conditional response time and eventually the mean response time of a job requiring T units of processing time will be given as follows.

Algorithm to simulate response time of a job requiring T units of processing time in a 2-terminal system.

Let w_o = amount of work time remains to accomplish for the tagged job.

c_o = amount of clock time accumulated towards the response time of the tagged job.

Step 1 : Set w_o = T and c_o = 0

Step 2 : If the tagged job enters to find the system empty, otherwise go to step 3, generate an exponential time with parameter $\lambda/r(1)$. Call this t'.

```
a) If $t' \geq w_0$, set the conditional response time
    $R_1 = c_0 + w_0/r(1)$.
    STOP
b) If $t' < w_0$, set :
    $w_0 = w_0 - t'$, $c_0 = c_0 + t'/r(1)$.
    GO TO Step3.
```

Step 3 : If the job enters to find another one already present in the system, i.e. $j = 2$, generate an exponential with parameter μ. Call this t''.

```
a) If $t'' \geq w_0$, set the response time
    $R_2 = c_0 + 2w_0/r(2)$.
    STOP
b) If $t'' < w_0$, set :
    $w_0 = w_0 - t''$, $c_0 = c_0 + 2t''/r(2)$.
    GO TO Step 2.
```

Repeat the above procedure until we obtain the conditional response times for both cases.

To find the mean response time we use the long-run distribution of the number of jobs in system just after the tagged job entered, i.e.

$$q_j = c\pi_{j-1}\lambda_{j-1} = c\pi_j \mu r(j),$$

where $j = 1,2$ and $q_1 + q_2 = 1$ and π_j is the stationary distribution of the continuous time Markov chain, $\{X(t)\}$.

The mean response time of a job requiring t units of work time is then

$$E[R] = R_1 q_1 + R_2 q_2,$$

where R_1, R_2 are the conditional mean response times, given the tagged job enters to find the system empty and one job already present respectively, generated by the algorithm.

The variance, skewness and kurtosis of the response time may also be obtained by deriving the usual central

moments. These calculations will be detailed in the next section.

2. Numerical Results

The numerical results shown in Table II are obtained from the outputs of a computer program written in FORTRAN .

TABLE II

Simulation results for 2 terminals

$\lambda = 1, \mu = 2$

Time	Mean	Std.Dev.	Skewness	Kurtosis
0.1000	0.1339 (.0006)	0.0452	0.6839	-1.4462
0.2000	0.2648 (.0014)	0.0851	0.7550	-1.2457
0.3000	0.4031 (.0027)	0.1229	0.6459	-1.3044
0.5000	0.6688 (.0047)	0.1913	0.6477	-1.1835
1.0000	1.3669 (.0098)	0.3200	0.6017	-1.0297
2.0000	2.6864 (.0157)	0.5051	0.5690	-0.4870
3.0000	4.0128 (.0211)	0.6516	0.5184	-0.2896
5.0000	6.6863 (.0269)	0.8224	0.2974	-0.1623

using the above algorithm to simulate the conditional response time, given the initial system state when the tagged job arrives. The steady state distribution is then used to evaluate the unconditional moments, and measures of skewness and kurtosis for various work time requirements of the job. The values between parentheses below the means are their corresponding standard errors which depend on the number of simulation replications. These outputs are evaluated based on 500 replications of the response time for each initial condition. The r(j)'s are all assumed to be unity.

Note that the moments obtained from the simulation agree well with those obtained by solving directly the system of differential equations for the moments of the response times. This fact provides a check for the simulation. The simulated response times show diminishing values of skewness and kurtosis as the required work time becomes large, again suggesting that there may be an increasingly accurate normal approximation to the response time distribution.

D. SIMULATION FOR AN N-TERMINAL SYSTEM

1. Algorithm

Now, consider a more general computer system with one processor and N terminals. As before, each terminal has a submission rate λ to the processor and the processor processes each job to completion with rate μ. The allocation of processing time always follows the method of "Processor-Sharing".

The simulation for response time of a job requiring T units of processing time will be done under the condition that when the job arrives there are j terminals active, i.e. j jobs, including the tagged one that just arrived, are being served by the CPU. Here, j can be 1,2,...,N.

Under Processor-Sharing scheduling, if j jobs are present then in a short time interval of length h the tagged job gets hr(j)/j units of work done. Thus, if W(t) is the amount of work done on tagged job by the time it has been in the system for t units of clock time, and if the number of jobs in system during this time t is j, then W(t) = tr(j)/j.

As long as W(t) is less than the required amount of work T for the tagged job, we will have to accumulate the amount of work done computed according to the number of jobs in system at that time. The conditional response time will be the clock time t for which the accumulated completed work time $\int_0^t W(t')dt'$ is equal to T.

Again it is convenient to measure time in terms of work time. The work time process, C(w), described in the previous section is a birth-death process with rates $\lambda(N-j)j = \lambda'_j$ and $\mu'_j = \mu(j-1)$, $1 \leq j \leq N$. The response time is simply $R(T) = \int_0^T C(w)dw$, if r(j) = 1.

To simulate the work time process, we generate two exponential times with parameters λ'_j and μ'_j respectively. The minimum of the two will indicate which event, arrival or departure, occurs first. If an arrival occurs first and the accumulated completed work time is still less than the requirement, T, the number of jobs being processed by the CPU, i.e. system state, will be j+1. Likewise, when a departure occurs first the number of jobs for next computation will be j-1.

The above observations allow us to construct an algorithm to perform a simulation for conditional response time and lead eventually to the estimation of statistics for the response time of a job requiring T units of processing time as follows.

Algorithm to simulate response time of a job requiring T units of processing time for an N-terminal system, given when the tagged job begins processing there are (j-1) jobs also being processed.

Let w_0 = amount of work that remains to accomplish for the tagged job.

c_0 = amount of clock time accumulated towards the response time of the tagged job.

```
Step 1 :  Set $w_0 = T$ and $c_0 = 0$.

Step 2 :  If $j=1$, i.e. the job enters to find the system
empty, otherwise GO TC step 3, generate an exponential time
with parameter $\lambda'_1$. Call this $t'$.
        a) If $t' \geq w_0$, set the response time
              $R_1 = c_0 + w_0/r(1)$
              STOP
        b) If $t' < w_0$, set
              $w_0 = w_0 - t'$
              $c_0 = c_0 + t'/r(1)$
              $j = 2$
              GO TO step 3.

Step 3 :  If $j = 2,3,\ldots,N-1$, generate two exponential times
with parameters $\lambda'_j$ and $\mu'_j$. Call them $t'$ and $t''$ respectively.
        a) If $\min(t',t'') \geq w_0$, set the response time
              $R_j = c_0 + w_0 j/r(j)$
              STOP.
        b) If $\min(t',t'') < w_0$:
              i) If $\min(t',t'') = t'$, set
                    $w_0 = w_0 - t'$
                    $c_0 = c_0 + t' j/r(j)$
                    $j = j + 1$
                    GO TO step 2 or 3 or 4 according to j.
              ii) If $\min(t',t'') = t''$, set
                    $w_0 = w_0 - t''$
                    $c_0 = c_0 + t'' j/r(j)$
                    $j = j - 1$
                    GO TO step 2 or 3 or 4 according to j.
```

Step 4 : If $j = N$, i.e. all terminals become active, generate an exponential time with parameter μ'_N. Call this t''.

```
    a) If $t'' \geq w_0$, set the response time
        $R_N = c_0 + w_0 N/r(N)$
        STOP
    b) If $t'' < w_0$, set
        $w_0 = w_0 - t''$
        $c_0 = c_0 + t'' N/r(N)$
        $j = N - 1$
        GO TO step 3.
```

Each run of the algorithm for fixed initial j gives a realization of the conditional response time of the tagged job given there are (j-1) other jobs in the system when the tagged job arrives for processing.

2. Moments of Response Time

The simulation based on the above algorithm provides a batch of conditional response times of a tagged job for each initial system state.

Suppose we simulate a batch of size K for each conditional response time at initial system state j, for all j's.

Let R_{jk} be the k-th realized conditional response time given the initial condition is j ; (that is, the tagged job arrives when (j-1) other jobs are being processed), for $j = 1,2,\ldots,N$ and $k = 1,2,\ldots,K$.

Mathematically, we can use the averages of $(R_{jk})^i$, $i = 1,2,3,4$, over each batch to compute empirical first, second, third and fourth conditional moments given initial condition j respectively. To compute the unconditional empirical moments for response time of a tagged job that requires T units of processing time, we can multiply the j-th conditional empirical moment by the steady-state

probability q_j given by (2.4) and sum over all $j = 1,2,\ldots,N$. The empirical mean, variance, third and fourth central moments then are computed by power expansions, i.e.

$$\begin{aligned}
\hat{E}[R(T)] &= \sum_{j=1}^{N} \bar{R}_j q_j = \bar{R} \\
\hat{Var}[R(T)] &= \sum_{j=1}^{N} \bar{R}_j^2 q_j - (\bar{R})^2 \qquad (3.1) \\
\hat{E}[(R(T)-R)^3] &= \sum_{j=1}^{N} \bar{R}_j^3 q_j - 3\bar{R}\sum_{j=1}^{N} \bar{R}_j^2 q_j + 2(\bar{R})^3 \\
\hat{E}[(R(T)-R)^4] &= \sum_{j=1}^{N} \bar{R}_j^4 q_j - 4\bar{R}\sum_{j=1}^{N} \bar{R}_j^3 q_j + 6(\bar{R})^2\sum_{j=1}^{N} \bar{R}_j^2 q_j - 3(\bar{R})^4,
\end{aligned}$$

where $\bar{R}_j^i = (\sum_{k=1}^{K} R_{jk})/K$, for $i = 1,2,3,4$.

However, in practice the above procedure can be numerically unstable since the averages of the second, third and fourth moments over the batch may be very large numbers. Hence, when we add or subtract these numbers to compute the central moments, it is possible that the computation produces round-off errors which may be substantial. Therefore, we would rather rewrite the central moments in terms of conditional expectations, i.e.

$$E[(R-E(R))^i] = E[E[(R-E(R))^i \mid X(0)]], \quad i=1,2,3,4. \qquad (3.2)$$

where $X(0)$ is the number of jobs requesting processing by the CPU, including the tagged job, when the tagged job arrives. This allows us to obtain the central moments in a more numerically stable manner. Further details will be shown for the computations of variance, skewness and kurtosis of the response time.

3. Computation of Simulated Variance

Let $\bar{R}_j = (\sum_{k=1}^{K} R_{jk})/K$ and $\bar{R} = \sum_{j=1}^{N} \bar{R}_j q_j$, then the estimated variance

$$\widehat{Var}[R] = \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R})^2 q_j$$

$$= \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R}_j+\bar{R}_j-\bar{R})^2 q_j$$

$$= \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}[(R_{jk}-\bar{R}_j)^2+2(R_{jk}-\bar{R}_j)(\bar{R}_j-\bar{R})+(\bar{R}_j-\bar{R})^2]q_j .$$

Since $(\sum_{k=1}^{K} R_{jk}-\bar{R}_j)/K = 0$, the estimated variance becomes

$$\widehat{Var}[R] = \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R}_j)^2 q_j + \sum_{j=1}^{N}(\bar{R}_j-\bar{R})^2 q_j .$$

This is the sampling version of the formula

$$Var[R] = E[Var(R|X(0))]+Var[E(R|X(0))],$$

which is a known general result that applies to any random variable, R, that also depends upon another random variable, namely X(0). The first component represents the overall variability of R for a fixed value of X(0), and the second component represents the variability of R due to the variability in X(0).

4. Computation of Simulated Skewness

To compute the estimated skewness we first compute the third estimated central moment of the response time by rewriting it in a conditional expectation form as we have done for the computation of the variance. The third central moment is derived as shown below.

$$\widehat{E}[(R_{jk}-\bar{R})^3] = \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R})^3 q_j$$

$$\widehat{E[}(R_{jk}-\bar{R})^3] = \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R}_j+\bar{R}_j-\bar{R})^3 q_j$$

$$= \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}[(R_{jk}-\bar{R}_j)^3+3(R_{jk}-\bar{R}_j)^2(\bar{R}_j-\bar{R})$$

$$+3(R_{jk}-\bar{R}_j)(\bar{R}_j-\bar{R})^2+(\bar{R}_j-\bar{R})^3]q_j.$$

We can remove the 3-rd component on the right hand side, since $(\sum_{k=1}^{K} R_{jk}-\bar{R}_j)/K = 0$. Thus

$$\widehat{E[}(R_{jk}-\bar{R})^3] = \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R}_j)^3 q_j + \frac{3}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R}_j)^2(\bar{R}_j-\bar{R})q_j$$

$$+\sum_{j=1}^{N}(\bar{R}_j-\bar{R})^3 q_j.$$

The measure of skewness of response time is then

$$\widehat{\text{Skewness}}[R_{jk}] = \frac{E[(R_{jk}-\bar{R})^3]}{(\text{Var}[R_{jk}])^{3/2}}.$$

5. Computation of Simulated Kurtosis

We start by rewriting the expression for the estimated fourth central moment as follows:

$$\widehat{E[}(R_{jk}-\bar{R})^4] = \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R})^4 q_j$$

$$= \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R}_j+\bar{R}_j-\bar{R})^4 q_j$$

$$= \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}[(R_{jk}-\bar{R}_j)^4+4(R_{jk}-\bar{R}_j)^3(\bar{R}_j-\bar{R})$$

$$+6(R_{jk}-\bar{R}_j)^2(\bar{R}_j-\bar{R})^2+4(R_{jk}-\bar{R}_j)(\bar{R}_j-\bar{R})^3$$

$$+(\bar{R}_j-\bar{R})^4]q_j.$$

We then simplify, as before, by removing the 4-th term on the right hand side. Hence,

$$\widehat{E}[(R_{jk}-\bar{R})^4] = \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R}_j)^4 q_j + \frac{4}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R}_j)^3(\bar{R}_j-\bar{R})q_j$$

$$+\frac{6}{K}\sum_{k=1}^{K}\sum_{j=1}^{N}(R_{jk}-\bar{R}_j)^2(\bar{R}_j-\bar{R})^2 q_j + \sum_{j=1}^{N}(\bar{R}_j-\bar{R})^4 q_j .$$

Therefore, the measure of kurtosis is

$$\widehat{\text{Kurtosis}}[R_{jk}] = \frac{E[(R_{jk}-\bar{R})^4]}{(\text{Var}[R_{jk}])^2} .$$

Since for the normal distribution the kurtosis has the value 3, we then substract 3 from the kurtosis computed above. So the new value of kurtosis will be 0 when the distribution of response time has the normal degree of kurtosis.

6. Standard Error of the Mean Response Time

In order to assess the accuracy of the simulated average response time, we may compute a standard error for the mean response time of a tagged job that requires T units of work time from our batches of simulated conditional response times.

We have derived previously that the mean response time is

$$\widehat{E}[R_{jk}] = \bar{R} = \sum_{j=1}^{N}\left(\frac{1}{K}\sum_{k=1}^{K}R_{jk}\right)q_j .$$

Now, we apply a property of the variance function by considering q and K fixed constants and noting that the R_{jk}'s are obtained from independent realizations. This allows us to write the estimated variance of the expected response time as follows :

$$\widehat{\text{Var}}[\widehat{E}[R_{jk}]] = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{1}{K}\sum_{k=1}^{K}(R_{jk}-\bar{R}_j)^2\right)q_j^2$$

To obtain the standard error of the mean response time we simply take the square root of $\widehat{Var}[\hat{E}[R_{jk}]]$.

7. Numerical Results

Tables III, IV and V show the outputs from a simulation program based on the previously described algorithm. The number of replications for each initial condition is

TABLE III

Numerical results from simulation

$N = 10, \lambda = 15, \mu = 100$

Time	Mean	Variance	Skewness	Kurtosis
0.0100	.040368 (.000177)	.000295	.119865	-.715162
0.0250	.100510 (.000503)	.001405	.088115	-.712345
0.0375	.151623 (.000733)	.002508	.015783	-.587745
0.0500	.201580 (.000952)	.003865	-.046461	-.572722
0.0625	.251340 (.001092)	.004953	-.012371	-.510326
0.1000	.402916 (.001510)	.008842	-.092651	-.378387
0.1500	.607487 (.001935)	.013806	-.061429	-.186003
0.2000	.806743 (.002303)	.019704	-.023100	-.130595

TABLE IV

Numerical results from simulation

$N = 10, \lambda = 25, \mu = 100$

Time	Mean	Variance	Skewness	Kurtosis
0.0100	.060567 (.000192)	.000253	-.773193	.685650
0.0250	.150692 (.000467)	.000992	-.717666	.637294
0.3750	.226364 (.000652)	.001775	-.697005	.532480
0.0500	.302138 (.000791)	.002466	-.662308	.538648
0.0625	.378130 (.000925)	.003266	-.649630	.527906
0.1000	.603584 (.001228)	.005601	-.538279	.509166

500. The numbers below the means are their standard errors. We can see that the results in the tables indicate a somewhat symmetric distribution for the response time (skewnesses are very small), and the kurtosises do not strongly indicate non-normality (they decrease towards zero), especially when the processing time requirement becomes large. The kurtosis values in table III seem to indicate smaller tails than those in table IV, and the skewness values indicate that we have a more symmetric distribution in the case $\lambda = 15$ than when $\lambda = 25$ for an equal $\mu = 100$. In the next chapter some normal approximations to the response time distribution will be described.

TABLE V

Numerical results from simulation

$N = 25, \lambda = 5, \mu = 100$

Time	Mean	Variance	Skewness	Kurtosis
0.0100	.063016 (.000202)	.001011	.346171	-.463719
0.0250	.158884 (.000637)	.005204	.213798	-.555656
0.0375	.239094 (.001008)	.010175	.170748	-.541305
0.0500	.316021 (.001354)	.015985	.159817	-.565249
0.0625	.394780 (.001686)	.021523	.117365	-.496104

$N = 25, \lambda = 10, \mu = 100$

Time	Mean	Variance	Skewness	Kurtosis
0.0100	.149847 (.000252)	.000659	-.527812	.471482
0.0250	.375932 (.000588)	.002529	-.644450	.428531
0.0375	.562173 (.000834)	.004470	-.667446	.366822
0.0500	.750556 (.001020)	.006467	-.641420	.314407
0.0625	.938072 (.001165)	.008259	-.593361	.275313

IV. NORMAL APPROXIMATION FOR RESPONSE TIME

A. INTRODUCTION

The Markov assumptions we make on the processor-sharing system allow us to infer that the distribution of the response time of a tagged job that requires T units of processing time may be approximated by the normal distribution when T is large and/or when the system is in heavy traffic.

Two methods are used to argue the approximate normality of the distribution of response time. One is based on the Central Limit Theorem for additive functionals of a birth and death process, and the other follows from a heavy-traffic diffusion approximation of the birth and death process. The formulas to compute the approximate mean and variance will be described. More details concerning the analytic form of the approximations are given by Gaver, Jacobs and Latouche. [Ref. 3] The approximations will be compared with the results from simulation to study their accuracy.

As mentioned before, the Central Limit Theorem for additive functionals of Markov processes allows the conclusion that the accumulated work accomplished in time t' of a job requiring a large amount of processing time, T, is approximately normally distributed. This in turn allows the conclusion that the corresponding response time is also approximately normally distributed. Hence, we will start by considering W(t'), the total work expended by the computer on the tagged job by clock time t' after its arrival, given that the tagged job requires exactly T time-units of work for completion.

It is observed that if when a job arrives there are $X(0) = j$ customers, including the new arrival, present in the system for processing, then:

$$W(t') = \int_0^{t'} (r(X_c(u))/X_c(u))\,du \;, \quad X_c(0) = j \geq 1, \qquad (4.1)$$

where $X_c(t)$ is the number of jobs at the service stage at clock time t.

From this an appropriate central limit theorem for W(t) can be established by using results for finite birth-and-death models [Ref. 8] or by making use of the theory of cumulative processes. [Ref. 9] We note here that the latter development of the central limit theorem is adaptable to models more general than the simple birth-and-death process.

In the case in which the system is in heavy traffic we can approximate $\{X_c(t); t \geq 0\}$ by an Ornstein-Uhlenbeck process, see e.g. Iglehart. [Ref. 10] The process W(t') is then approximated by an integral of an Ornstein-Uhlenbeck process. A normal approximation for the response time distribution then follows.

After deriving some formulas for the above two methods of approximation, we will make comparison for goodness of fit of those approximations to simulation data.

B. APPROXIMATION BY CENTRAL LIMIT THEOREM

1. A Central Limit Theorem for W(t)

Under Markov assumptions throughout the processor-sharing system, the number of jobs being at service stage at clock time t, $X_c(t)$, is a finite ergodic stationary time reversible Markov chain.

If we define a function $f(X(t)) = r(X(t))/X(t)$, then as outlined in Keilson [Ref. 8] the process in equation (4.1) may be proven in a variety of ways to be asymptotically normal in distribution for large t. Hence, the accumulated accomplished work-time at time t, W(t), satisfies a central limit theorem.

In order to apply the central limit theorem derived by Keilson we will have to redefine the infinitesimal generator according to the number of jobs in system without including the tagged one.

The relevant generator is now

$$X'(t) = j \text{ ---> } X'(t+h) = j+1 \; : \; \lambda_{j+1}h + o(h) \qquad (4.2)$$

$$\text{---> } X'(t+h) = j-1 \; : \; \mu_{j+1}hj/(j+1) + o(h)$$

$$\text{---> } X'(t+h) = j \; : \; 1-(\lambda_{j+1} + \mu_{j+1}j/(j+1))h + o(h),$$

for $j = 0, 1, \ldots, N' = N-1$.

The process in equation (4.1) then becomes

$$W(t) = \int_0^t f(X'(u))\,du, \qquad (4.3)$$

where $f(X'(u)) = r(X'(u)+1)/(X'(u)+1)$.

We can now express a statement for the central limit theorem as :

$$\frac{W(t) - \zeta t}{\sigma\sqrt{t}} \text{ ---> } N(0,1), \text{ as } t \text{ ---> } \infty \qquad (4.4)$$

where the constants ζ and σ^2 are such that

$$\zeta = \sum_{j=0}^{N'} f(j)\pi'_j \qquad (4.5)$$

$$\sigma^2 = 2[f(0), f(1), \ldots, f(N')] \begin{bmatrix} \pi'_0 & & & \\ & \pi'_1 & & 0 \\ & & \ddots & \\ 0 & & & \pi'_{N'} \end{bmatrix} \underset{\sim}{Z} \begin{bmatrix} f(0) \\ f(1) \\ \vdots \\ f(N') \end{bmatrix} \qquad (4.6)$$

with $\underset{\sim}{Z}$ being a matrix defined below

$$\underset{\sim}{Z} = ([I-\underset{\sim}{A}+\underset{\sim}{L}]^{-1} - \underset{\sim}{L})/\gamma , \qquad (4.7)$$

I is an identity matrix, and $\underset{\sim}{L}$ is an N by N matrix whose rows are steady-state probabilities, $\{\pi'_j\}$, of the birth and death process with rates in (4.2), i.e.

$$\underset{\sim}{L} = \begin{bmatrix} \pi'_0 & \pi'_1 & \cdots & \pi'_{N'} \\ \pi'_0 & \pi'_1 & \cdots & \pi'_{N'} \\ \vdots & & & \\ \pi'_0 & \pi'_1 & \cdots & \pi'_{N'} \end{bmatrix} \qquad (4.8)$$

and $\underset{\sim}{A}$ is a matrix defined as follows :

$$A_{0,1} = \lambda_1/\gamma , \quad A_{0,0} = 1-\lambda_1/\gamma$$

$$A_{j,j+1} = \lambda_{j+1}/\gamma, \quad A_{j,j-1} = \mu_{j+1}(j/(j+1))/\gamma, \quad A_{j,j} = 1-\nu_j/\gamma \qquad (4.9)$$

$$\nu_j = \lambda_{j+1} + \mu_{j+1}(j/(j+1)), \quad \gamma = \max_j \nu_j$$

$(\nu_0 = \lambda_1, \ \nu_{N'} = \mu_N(N'/N))$, (again with rates in (4.2)).

2. A Central Limit Theorem for response time, R(T)

The accumulated accomplished work-time at clock time t' of a tagged job requiring T units of processing time, W(t'), increases in random straight-line segments from W(0) = 0 until W(t') = T. The value of t' here will be the response time of the job, R(T). It is the first-passage time to the required work time. So we can state the following:

$$P(W(t') < T) = P(R(T) > t'). \qquad (4.10)$$

From equations (4.4) and (4.10), we can derive, by the same standard argument of renewal theory as given in Karlin and Taylor [Ref. 11] pp. 208-209, the following statement of a central limit theorem for response time, R(T), as T approaches infinity, i.e.

$$\frac{R(T) - (T/\xi)}{\sqrt{T\sigma^2/\xi^3}} \longrightarrow N(0,1), \text{ as } T \longrightarrow \infty. \qquad (4.11)$$

C. APPROXIMATION BY HEAVY TRAFFIC ANALYSIS

Now, we consider the processor-sharing system as a machine repair model in which $\lambda_j = \lambda(N-j)$, $\mu_j = \mu$, and $r(j) = 1$ for all j's. Let N the number of terminals be large and the traffic intensity ρ, which is defined as the ratio of the expected service time to N times the expected inter-arrival time, be a fixed value less than one, i.e. $\rho = \mu/(N\lambda)$.

Under the above conditions, when a job requiring T units of processing time enters the system it will be processed in the company of many others. This indicates a system with heavy traffic situation. As mentioned earlier in the introduction section we may apply the properties of the limiting diffusion process developed by Iglehart. [Ref. 10]

Therefore, if $X_c(t)$ is the number of jobs at the processing stage at clock time t, then $X_c(t)$ can be approximated by a diffusion process as follows:

$$X_c(t) = Na(t) + \sqrt{N}.Y(t), \qquad (4.12)$$

where $a(t)$ is a deterministic function of time and when t approaches infinity it has a finite limit, i.e. $a(\infty) = 1-\rho$, and $\{Y(t)\}$ is, for the present model, a particular Ornstein-Uhlenbeck process.

The accumulated completed work-time by fixed time t' of a tagged job requiring T units of processing time in equation (4.1) becomes

$$W(t') = \int_0^{t'} du/X_c(u) = \int_0^{t'} du/(Na(u)+\sqrt{N}.Y(u)). \qquad (4.13)$$

Next, we apply the approximation and expand the expression in (4.13) to second order terms in N; then assume that the tagged job arrives when the system is in steady state so that we can use the finite limit of $a(\infty) = 1-\rho$ in place of $a(u)$. The approximation is now:

$$W(t') = \int_0^{t'} du/(N(1-\rho)) - (\sqrt{N}/[N(1-\rho)]^2) \int_0^{t'} Y(u)du, \qquad (4.14)$$

for $0 \le t' \le t$.

Hence, the expectation of accumulated amount of work-time completed on the tagged job is approximately $t'/(N(1-\rho))$, and the actual distribution of total work done is also approximately Gaussian (integral of an Ornstein-Uhlenbeck process), where the Gaussian property results from the assumption of many accompanying jobs, and not necessarily because the tagged job requires a long processing time.

As t' approaches infinity, we can evaluate (4.14) and show that $E[\int_0^{t'} Y(u)du] = 0$, and $Var[\int_0^{t'} Y(u)du] = (2\mu/(N\lambda^2))t'$. So the normal approximation to accumulated completed work time, $W(t')$, has the parameters:

$$\xi = 1/(N(1-\rho)), \quad \sigma^2 = 2\mu/(\lambda^2[N(1-\rho)]^4).$$

By reasoning the same way as to obtain a central limit theorem for response time from a known approximated normal distribution of accumulated completed work-time for a tagged job requiring T units of processing time, we can derive a normal approximation for the distribution of response time as having the parameters as follow:

$$E[R(T)] = N(1-\rho)T, \tag{4.15}$$

$$Var[R(T)] = 2\mu T/(\lambda^2 N(1-\rho)). \tag{4.16}$$

We can also improve the value of the variance by using the formula below:

$$Var[R(T)] = [2\mu T/(\lambda s)]\delta , \tag{4.17}$$

where $s = \lambda N - \mu$ and $\delta = 1 - [(1 - e^{-sT})/(sT)]$.

D. COMPARISON TO SIMULATION DATA

We will consider some particular simulation results from a system consisting of 10 terminals and one processor in the cases of light traffic and heavy traffic to make comparison to normal approximations described in previous section.

First we can use the measures of skewness and kurtosis of the response time resulting from simulation data to roughly assess the degree of normality or non-normality. We know that if a distribution is symmetric its skewness will be zero and for the normal distribution its kurtosis has the value 3. However, even if a distribution of the simulated

response time has the measures of skewness and kurtosis close to those of the normal distribution, it does not imply that the distribution is necessarily normal. It only suggests that a normal distribution may be a reasonable approximation.

Secondly, we will compute the empirical distribution of the simulation data at various quantiles of the normal distribution whose mean and variance are approximated by either central limit theorem or heavy traffic analysis (limiting diffusion). For example, to compare the one-tenth quantile of the approximated normal distributions to the simulated response time at the initial system state j, we first compute

$$R_{.10} = \text{Approx. Mean} + (\text{Approx. Std. Dev.})(Z_{.10})$$

where $Z_{.10}$ is the (1/10)th quantile of the standard normal distribution. We then determine the conditional relative frequency of the simulated response time, given the initial system state is j, as

$$\text{r.f.}_j(R_{.10}) = \frac{\text{No. of simulated response time} \le R_{.10}.}{\text{No. of replications}}$$

We then use the distribution q_j in (2.4) to remove the initial state condition. Hence, we obtain the one-tenth quantile of the simulated response time (corresponded to the normal approximations). We may also compute the standard error of the estimated quantile by taking the square root of

$$\text{Var}(\hat{p}) = \Big(\sum_{j=1}^{N} q_j^2 \hat{p}_j (1-\hat{p}_j)\Big)/K,$$

where $\hat{p}_j = \text{r.f.}_j(R_{.10})$ and K = number of replications used in simulation.

Now consider a system of 10 terminals with arrival rate $\lambda = 15$ and service rate $\mu = 100$. The expected number of active terminals is, therefore,

$$E[X(t)] = 10(1-(100/150)) = 3.3333 ,$$

which indicates a light traffic situation.

TABLE VI
Comparison of Simulation Data to Normal Quantiles

$N = 10, \lambda = 15, \mu = 100$

Time	Normal	.10	.25	.50	.75	.90	.95	.99
0.0100	CLT	0.0	.123	.503	.887	.995	1.0	1.0
	HTA I	0.0	0.0	.361	.948	1.0	1.0	1.0
	HTA II	0.0	.102	.361	.677	.909	.972	.999
0.0250	CLT	.042	.211	.487	.806	.967	.994	1.0
	HTA I	0.0	.012	.334	.832	.996	1.0	1.0
	HTA II	0.0	.094	.334	.668	.914	.977	1.0
0.0500	CLT	.083	.232	.486	.766	.938	.977	.999
	HTA I	0.0	.032	.300	.737	.970	.998	1.0
	HTA II	0.0	.067	.300	.649	.916	.973	1.0
0.0750	CLT	.082	.241	.494	.766	.922	.967	.997
	HTA I	0.0	.033	.266	.688	.948	.991	1.0
	HTA II	.001	.049	.266	.630	.900	.969	1.0
0.1000	CLT	.097	.243	.483	.761	.919	.969	.998
	HTA I	.000	.027	.238	.647	.934	.987	1.0
	HTA II	.001	.037	.238	.603	.893	.972	1.0

TABLE VII

Comparison of Simulation Data to Normal Quantiles

$N = 10,\ \lambda = 25,\ \mu = 100$

Time	Normal	.10	.25	.50	.75	.90	.95	.99
0.0100	CLT	.042	.153	.447	.854	.997	1.0	1.0
	HTA I	.048	.158	.434	.826	.990	1.0	1.0
	HTA II	.106	.224	.434	.721	.911	.975	1.0
0.0250	CLT	.077	.191	.453	.804	.975	.998	1.0
	HTA I	.084	.192	.438	.774	.955	.993	1.0
	HTA II	.104	.216	.438	.729	.917	.976	1.0
0.0500	CLT	.092	.215	.454	.769	.951	.990	1.0
	HTA I	.095	.212	.435	.728	.927	.979	1.0
	HTA II	.108	.224	.435	.707	.909	.965	.998
0.0750	CLT	.093	.198	.447	.754	.936	.983	1.0
	HTA I	.095	.193	.419	.713	.908	.967	.998
	HTA II	.102	.200	.419	.701	.892	.959	.996
0.1000	CLT	.101	.236	.474	.756	.933	.976	.999
	HTA I	.102	.226	.448	.711	.898	.958	.997
	HTA II	.106	.232	.448	.704	.891	.953	.996

Table III shows the values of skewness and normal kurtosis of the response time of a tagged job for various work-time requirements. Those values are quite small especially for large work-time requirements. It indicates the possibility of a successful normal approximation.

Table VI shows how the simulated response time of a job requiring some processing time units is distributed

comparing to the various quantiles (.10, .25, .50, .75, .90, .95, .99) of the normal distributions approximated by central limit theorem and the two heavy traffic (limiting diffusion) approaches (HTA I is (4.16), HTA II is (4.17)).

We see that in this situation the limiting diffusion approximation for normality of the distribution of response time is not so good an approach even for large amount of processing time requirement. This is easily anticipated since the system has only an expected proportion of one-third of its terminals active. However, the central limit theorem seems to work pretty well when the required work time becomes very large.

Let's consider a new system with arrival rate $\lambda = 25$ and the service rate $\mu = 100$, with $N = 10$ again. The expected number of active terminals is now

$$E[X(t)] = 10(1-(100/250)) = 6,$$

which indicates a moderately heavy traffic in the system.

Table IV provides us some good feelings about the measures of skewnesses and normal kurtosises not being too far from those of actual normal distribution. Now, observe the table VII which shows the distribution of the simulated response time of a tagged job for various work time requirements comparing to the various quantiles of the normal distributions approximated by central limit theorem and limiting diffusion approaches.

We see that the limiting diffusion methods, in this case, work pretty well even for small work-time requirements. The one with second formula (4.17) of approximating the variance works better than the other, because it provides a smaller standard deviation. The central limit theorem approximation still works very well when the work time requirement becomes large.

TABLE VIII

Comparison of Simulation Data to Normal Quantiles

$N = 25, \lambda = 10, \mu = 100$

Time	Normal	.10	.25	.50	.75	.90	.95	.99
0.0100	CLT	.030	.150	.456	.859	.987	.999	1.0
	HTA I	.047	.161	.456	.845	.980	.998	1.0
	HTA II	.109	.233	.456	.727	.908	.965	.997
0.0250	CLT	.070	.192	.460	.799	.961	.992	1.0
	HTA I	.082	.204	.460	.780	.948	.986	1.0
	HTA II	.112	.233	.460	.730	.910	.967	.997
0.0375	CLT	.087	.212	.470	.784	.948	.986	1.0
	HTA I	.097	.226	.470	.763	.934	.978	.999
	HTA II	.115	.243	.470	.731	.910	.964	.997
0.0500	CLT	.089	.210	.462	.778	.940	.982	.999
	HTA I	.099	.223	.462	.760	.927	.976	.999
	HTA II	.111	.236	.462	.734	.910	.963	.997
0.0625	CLT	.090	.220	.467	.772	.933	.977	.999
	HTA I	.100	.233	.467	.751	.917	.968	.998
	HTA II	.112	.244	.467	.727	.903	.960	.997

$N = 40, \lambda = 10, \mu = 100$

Time	Normal	.10	.25	.50	.75	.90	.95	.99
0.0100	CLT	.072	.198	.477	.789	.954	.988	1.0
	HTA I	.076	.203	.477	.784	.949	.986	1.0
	HTA II	.110	.243	.477	.739	.902	.961	.997
0.0375	CLT	.094	.224	.477	.749	.917	.966	.997
	HTA I	.098	.227	.477	.743	.912	.963	.996
	HTA II	.106	.236	.477	.736	.900	.958	.995
0.0625	CLT	.097	.232	.482	.748	.917	.965	.997
	HTA I	.100	.236	.482	.744	.909	.962	.997
	HTA II	.105	.241	.482	.737	.903	.957	.995

The above results for the two situations, one light traffic and another moderately heavy traffic, lead to the confirmation that for a tagged job requiring large amount of processing time the distribution of response time approaches, asymptotically, normality. In heavy traffic case the limiting diffusion approximations seem to work better than the central limit theorem approach. They do not work well at all when the system has the terminals active less than a half of its full capacity.

To conclude this chapter we show some more results in comparing the approximated normal quantiles to the values of simulated response times in table VIII for different (larger) number of terminals and various transitional rates. We can see easily that as the number of terminals is large the distribution of simulated response time approaches the approximated normals more rapidly. The heavy traffic approximation seems to work better, in these cases, than the central limit theorem approach, especially for small required work-time. This corresponds to our observation that the central limit theorem requires large work-time requirement to be a good approximation, while the only condition that the heavy traffic approximations require is to have a certain amount of jobs waiting to be served at any instant. Finally, table IX shows the mean and standard deviation of the response time computed by the central limit theorem and heavy traffic approximations for all the cases that we have been studying the comparison of the quantiles.

TABLE IX

Approx. Mean and Std. Dev. for One-Type Model

$N = 10$, $\lambda = 15$, $\mu = 100$

	CLT		HTA I		HTA II	
Time	Mean	Std.Dev	Mean	Std.Dev.	Mean	Std.Dev.
0.0100	.0404	.0323	.0333	.0516	.0333	.0238
0.0250	.1010	.0510	.0833	.0816	.0833	.0535
0.0500	.2019	.0722	.1667	.1155	.1667	.0919
0.0750	.3029	.0884	.2500	.1414	.2500	.1216
0.1000	.4038	.1020	.3333	.1633	.3333	.1462
$N = 10$, $\lambda = 25$, $\mu = 100$						
0.0100	.0605	.0245	.0600	.0231	.0600	.0160
0.0250	.1513	.0388	.1500	.0365	.1500	.0314
0.0500	.3027	.0548	.3000	.0516	.3000	.0481
0.0750	.4540	.0672	.4500	.0632	.4500	.0604
0.1000	.6053	.0776	.6000	.0730	.6000	.0706
$N = 25$, $\lambda = 10$, $\mu = 100$						
0.0100	.1500	.0390	.1500	.0365	.1500	.0254
0.0250	.3750	.0617	.3750	.0577	.3750	.0497
0.0375	.5625	.0755	.5625	.0707	.5625	.0641
0.0500	.7500	.0872	.7500	.0816	.7500	.0760
0.0625	.9375	.0975	.9375	.0913	.9375	.0863
$N = 40$, $\lambda = 10$, $\mu = 100$						
0.0100	.3000	.0264	.3000	.0258	.3000	.0213
0.0375	1.1250	.0512	1.1250	.0500	1.1250	.0477
0.0625	1.8750	.0661	1.8750	.0646	1.8750	.0628

V. MODEL FOR A SYSTEM WITH TWO TYPES OF TERMINALS

A. INTRODUCTION

The processor-sharing model we have been describing so far deals with a computer system having a single exponential service time distribution only. To generalize it we will consider a model of a computer system that consists of one Central Processing Unit, M terminals of type I and N terminals of type II. Type I terminals submit jobs to be processed by the processor at rate λ_1, which need an expected amount of work $1/\mu_1$. Likewise for type II terminals, the arrival rate to the processor is λ_2, and the expected work needed is $1/\mu_2$. Think times and amounts of work requested are to be independent and exponential.

This computer system may be viewed as the one having the ability to process two types of jobs. Each type of jobs must be submitted from its corresponding type of terminal.

The expected response time of a tagged job that requires T units of processing time will be derived by the same approach we used for the one-type model. However, now, we have to consider conditioning the given tagged job to be either type I or type II. The continuous time Markov-chain to be considered for the model is bivariate, with one variable being the number of jobs of type I being processed and the other being the number of jobs of type II being processed. The response time of the tagged job will depend on the numbers of both types of jobs in the system with it.

Once we obtain the expected conditional response time given the initial condition of how many others as described above, we will have to remove those conditions by the steady-state distribution of the tagged job given it is of

type I or of type II. The application of this steady state distribution to the conditional response time will enable us to compute the mean, standard deviation, and higher moments of response time of a tagged job that requires T units of work time, and hence the skewness and kurtosis as well. These values can help judge the goodness of the normal approximation.

Simulation will be used to generate the conditional response times as before. The steady state distribution of the tagged job will be used to compute estimates of the mean, variance, skewness and kurtosis of response time for a tagged job requiring T units of processing time. Finally, we will describe an approach for normal approximations by the central limit theorem, and by a heavy traffic approach as for the one-type model, and study their accuracy through simulation.

B. STEADY-STATE DISTRIBUTION

When the system has two types of terminals to deal with, the direct derivation to find differential equations or to apply the Laplace transforms for the conditional response time of a tagged job is much more complicated than when the system has only one type of terminal. Since, we now have to consider not only the condition that the job is tagged to find how many of them already present in the system but also the condition that the job is of what type, I or II.

However, under the processor-sharing concept with Markov assumptions, exponential think times and work request times, it will not be too hard to simulate the expected conditional response time, given that the job is one of the two types, and that it initially finds i-1 others of the same type and j of the other type present in the system. But then to remove those conditions we will have to apply the joint

steady-state distributions of the number of jobs present in the system just after the tagged job enters, given the tagged job is of specified type. The similar computation is performed for the case the tagged job is of another type.

Let M and N be the number of type I and type II terminals respectively. If $\widetilde{X_k}(t)$ is the number of type k jobs at clock time t for k = 1,2, then, before a tagged job enters and let (i,j), for i = 0,1,2, ... ,M and j = 0,1,2, ... ,N, be the system state i type I jobs and j type II jobs being processed, the limiting distribution of the number of type I and type II jobs in the system is

$$\widetilde{\pi}(i,j) = \lim_{t\to\infty} P[(\widetilde{X_1}(t), \widetilde{X_2}(t)) = (i,j)].$$

Consider the local balance equations for type I tagged jobs:

$$\begin{aligned}
\widetilde{\pi}(0,0)\, M\lambda_1 &= \mu_1 \widetilde{\pi}(1,0) \\
\widetilde{\pi}(1,0)\,(M-1)\lambda_1 &= \mu_1 \widetilde{\pi}(2,0) \\
&\vdots \\
\widetilde{\pi}(i,0)\,(M-i)\lambda_1 &= \mu_1 \widetilde{\pi}(i+1,0) \qquad (5.1)
\end{aligned}$$

which imply that, for i = 0,1,2,...,M, the probability that the system is in state (i,0) is

$$\widetilde{\pi}(i,0) = M(M-1)\ldots(M-i+1)\,(\lambda_1/\mu_1)^i\, \widetilde{\pi}(0,0). \qquad (5.2)$$

Likewise, we can derive the same thing for type II tagged jobs:

$$\begin{aligned}
\widetilde{\pi}(0,0)\,N\lambda_2 &= \mu_2\widetilde{\pi}(0,1) \\
\widetilde{\pi}(0,1)\,(N-1)\lambda_2 &= \mu_2\widetilde{\pi}(0,2) \\
&\vdots \\
\widetilde{\pi}(0,j)\,(N-j)\lambda_2 &= \mu_2\widetilde{\pi}(0,j+1)
\end{aligned} \tag{5.3}$$

which in turn imply that, for $j = 0,1,\ldots,n$,

$$\widetilde{\pi}(0,j) = N(N-1)\ldots(N-j+1)\,(\lambda_2/\mu_2)^j\,\widetilde{\pi}(0,0). \tag{5.4}$$

Other local balance equations are :

$$\begin{aligned}
\widetilde{\pi}(i,0)\,N\lambda_2 &= \frac{1}{i+1}\mu_2\widetilde{\pi}(i,1) \\
\widetilde{\pi}(i,1)\,(N-1)\lambda_2 &= \frac{2}{i+2}\mu_2\widetilde{\pi}(i,2) \\
\widetilde{\pi}(i,2)\,(N-2)\lambda_2 &= \frac{3}{i+3}\mu_2\widetilde{\pi}(i,3) \\
&\vdots \\
\widetilde{\pi}(i,j)\,(N-j)\lambda_2 &= \frac{(j+1)}{i+j+1}\mu_2\widetilde{\pi}(i,j+1).
\end{aligned} \tag{5.5}$$

Finally we can comfortably guess, from the equations (5.5), the steady-state distribution of the total number of jobs (of both types) before the tagged job enters as, for $i = 0,1,\ldots,M$ and $j = 0,1,\ldots,N$,

$$\widetilde{\pi}(i,j) = \binom{i+j}{j} M(M-1)\ldots(M-i+1)\,(\lambda_1/\mu_1)^i$$

$$\times\ N(N-1)\ldots(N-j+1)\,(\lambda_2/\mu_2)^j\ \widetilde{\pi}(0,0)\,. \qquad (5.6)$$

This distribution also satisfies the full balance equations. The steady-state distribution of the number of jobs in the system can be found by choosing $\widetilde{\pi}(0,0)$ so that $\sum_{i=0}^{M}\sum_{j=0}^{N}\widetilde{\pi}(i,j) = 1$. More details can be found in Gaver and Jacobs. [Ref. 12]

The steady state distribution of the entering tagged job being of type I and there being i jobs of type I and j jobs of type II processing when the tagged job enters is

$$q(i,j,I) = k\,\widetilde{\pi}(i-1,j)\,(M-i)\lambda_1, \qquad (5.7)$$

and for type II tagged job

$$q(i,j,II) = k\,\widetilde{\pi}(i,j-1)\,(N-j)\lambda_2, \qquad (5.8)$$

where k is selected so that $\sum_{i=0}^{M}\sum_{j=0}^{N}[q(i,j,I)+q(i,j,II)] = 1$.

Similarly, the conditional distribution of there being i jobs of type I and j jobs of type II processing when the tagged job arrives given the tagged job is of type I is

$$q_I(i,j) = k_I\,\widetilde{\pi}(i-1,j)\,(M-i)\lambda_1, \qquad (5.9)$$

where k_I is chosen so that $\sum_{i=0}^{M}\sum_{j=0}^{N} q_I(i,j) = 1$.

C. A CENTRAL LIMIT THEOREM FOR THE RESPONSE TIME

In this section we will present a central limit theorem for the conditional distribution of the response time of a tagged job requiring T units of work given it is of a particular type. More details can be found in Gaver and Jacobs. [Ref. 12]

In what follows we will assume the tagged job is of type I. Let X (t) be the number of type I jobs (excluding the tagged one) being processed at work time t ; that is when the tagged job has acquired t units of work. Then $\{(X_1(t), X_2(t));\ t\ge 0\}$ is a continuous time Markov chain with rates

$$(i,j) \;\text{--->}\; (i+1,j) \;:\; (M-(i+1))\lambda_1(i+j+1), \quad 0\le i<M-1 \tag{5.10}$$

$$(i,j) \;\text{--->}\; (i,j+1) \;:\; (N-(j+1))\lambda_2(i+j+1), \quad 0\le j<N \tag{5.11}$$

$$(i,j) \;\text{--->}\; (i-1,j) \;:\; i\mu_1, \qquad 1\le i\le M-1 \tag{5.12}$$

$$(i,j) \;\text{--->}\; (i,j-1) \;:\; j\mu_2, \qquad 1\le j\le N. \tag{5.13}$$

Similar arguments to those used in deriving (5.6) show that the limiting distribution $\pi(i,j)$ for the Markov chain having rates (5.10), (5.11), (5.12) and (5.13) is of the form

$$\pi(i,j) = \binom{i+j}{j}(M-1)(M-2)\dots(M-i)(\lambda_1/\mu_1)^i \tag{5.14}$$

$$\times\ N(N-1)\dots(N-j+1)(\lambda_2/\mu_2)^j\ \pi(0,0),$$

where $\pi(0,0)$ is chosen so that $\sum_{i=0}^{M}\sum_{j=0}^{N}\pi(i,j) = 1$.

The response time of the tagged job which requires work T is

$$R(T) = \int_{0}^{T}[X_1(u)+X_2(u)+1]du, \qquad (5.15)$$

an integral of a function of a continuous time Markov chain.

As a result the central limit theorem of Keilson [Ref. 8] applies to give that as T ---> ∞, there are constants m and σ such that

$$\frac{R(T) - mT}{\sigma\sqrt{T}}$$

has an asymptotic normal distribution with mean zero and variance one. In this case

$$m = \sum_{i=0}^{M-1}\sum_{j=0}^{N}(i+j+1)\,\pi(i,j). \qquad (5.16)$$

The variance σ^2 is computed as follows.

Let Q be the infinitesimal generator of $(X_1(t),X_2(t))$ with system states in the order, i.e. (0,0), ..., (M-1,0),(0,1), ... ,(M-1,1), ... ,(0,N), ... ,(M-1,N). Then Q is an Mx(N+1) square matrix having the form

$$Q = \begin{bmatrix} A_0 & R_0 & 0 & 0 & 0 & \cdots \\ M_1 & A_1 & R_1 & 0 & 0 & \cdots \\ 0 & M_2 & A_2 & R_2 & 0 & \cdots \\ 0 & 0 & M_3 & A_3 & R_3 & \cdots \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdots \end{bmatrix}$$

where the non-zero elements are the rates of those system states.

Let γ be the maximum absolute value of a diagonal element of Q, then by uniformization of the chain we obtain a matrix $\tilde{A} = I + \frac{1}{\gamma} Q$.

Let $f(i,j) = i+j+1$, $\pi_j(i) = \pi(i,j)$, $f_j(i) = f(i,j)$ and $(\pi f)_j = \sum_{i=0}^{M-1} \pi_j(i) f_j(i)$. Then according to Keilson [Ref. 8] the central limit theorem variance term, σ^2, for the integral $\int_0^T f(X_1(u), X_2(u))\,du$ is as follows.

$$\sigma^2 = 2[(\pi f)_0, (\pi f)_1, \ldots, (\pi f)_N]\, \underset{\sim}{Z} \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_N \end{bmatrix} \tag{5.17}$$

where $\underset{\sim}{Z} = \frac{1}{\gamma}[I - \underset{\sim}{A} + \underset{\sim}{L}]^{-1} - \underset{\sim}{L}$, and

$$\underset{\sim}{L} = \begin{bmatrix} \pi_0 & \pi_1 & \ldots & \pi_N \\ \pi_0 & \pi_1 & \ldots & \pi_N \\ \vdots & & & \\ \pi_0 & \pi_1 & \ldots & \pi_N \end{bmatrix} \tag{5.18}$$

If we define $\underline{X} = (x_0, x_1, \ldots, x_N) = [I - \underset{\sim}{A} + \underset{\sim}{L}]^{-1} f$, then, since $I - \underset{\sim}{A} = -\frac{1}{\gamma} Q$, we have

$$[-\frac{1}{\gamma} Q + \underset{\sim}{L}]\underline{X} = f\ . \tag{5.19}$$

Multiplying both sides of (5.19) by $\underline{\pi}$, and since $\underline{\pi} Q = 0$ and $\underset{\sim}{L}\underline{X} = \underline{\pi}\,\underline{X}$, we then have $\underline{\pi}\,\underline{X} = \underline{\pi} f$. Thus (5.19) becomes

$$-\frac{1}{\gamma} Q\underline{X} = f - (\underline{\pi} f), \tag{5.20}$$

or, better,

$$Q\underline{X} = -\gamma(f - \underline{\Pi}f). \tag{5.21}$$

We can then solve equation (5.21) for X and thus the central limit theorem variance σ^2 from equation (5.17).

D. HEAVY TRAFFIC APPROXIMATION FOR THE RESPONSE TIME

As before, consider a computer system consisting of M type I terminals with arrival rate λ_1 and service rate μ_1, and N type II terminals with arrival rate λ_2 and service rate μ_2 respectively. In this section we will present a heavy traffic approximation for the response time. More details of the approximation are described by Gaver and Jacobs. [Ref. 12]

Let $L_1 = \lambda_1 M$, $L_2 = \lambda_2 N$ and $c = M/N$. To simplify the notations used in deriving heavy traffic mean and variance, we define the following expressions.

First we solve for m_1 which is the positive solution of the quadratic equation;

$$\begin{aligned} 0 = {} & -L_2 L_1 c + (\mu_1 - L_1)\mu_2 L_1 \\ & + m_1[L_1{}^2\mu_2 + (\mu_1 - L_1)(\mu_1 L_2 - \mu_2 L_1) + cL_2 L_1 \mu_1] \\ & + m_1{}^2[L_1(\mu_1 L_2 - \mu_2 L_1)], \end{aligned} \tag{5.22}$$

then set $m_2 = [m_1\mu_1/(L_1(1-m_1))] - m_1$.

$$\begin{aligned} \text{Let } a_1 &= -[L_1(1-m_1) - \mu_1 - L_1(m_1+m_2)] \\ a_2 &= -[L_1(1-m_1)] \\ b_1 &= -[L_2(c-m_2)] \\ b_1 &= -[L_2(c-m_2) - \mu_2 - L_2(m_1+m_2)] \\ \sigma_1^2 &= L_1(1-m_1)(m_1+m_2) + m_1\mu_1 \end{aligned}$$

$$\sigma_2^2 = L_2(c-m_2)(m_1+m_2)+m_2\mu_2.$$

Now put $S_0 = (-(b_2+a_1)+\sqrt{(b_2+a_1)^2-4(a_1b_2-a_2b_1)})/2$

and $S_1 = (-(b_2+a_1)-\sqrt{(b_2+a_1)^2-4(a_1b_2-a_2b_1)})/2$

(Note that S and S are solutions to a quadratic equation).

Also put $\beta_{11} = -(S_0+b_2)/(S_1-S_0)$; $\beta_{21} = b_1/(S_1-S_0)$

$$\beta_{22} = -(S_0+a_1)/(S_1-S_0) \ ; \ \beta_{12} = a_2/(S_1-S_0)$$

$$K_{11} = (S_1+b_2)/(S_1-S_0) \ ; \ K_{12} = -a_1/(S_1-S_0)$$

$$K_{22} = (S_1+a_1)/(S_1-S_0) \ ; \ K_{21} = -b_1/(S_1-S_0)$$

$$\gamma_{31} = (b_2-b_1)/(a_1b_2-a_2b_1) = (b_2-b_1)/(S_0S_1)$$

$$\gamma_{32} = (a_1-a_2)/(a_1b_2-a_2b_1) = (a_1-a_2)/(S_0S_1)$$

$$\beta_{31} = (\gamma_{31}S_1+1)/(S_0-S_1) \ ; \ \beta_{32} = (\gamma_{32}S_1+1)/(S_0-S_1)$$

$$K_{31} = -(\gamma_{31}S_0+1)/(S_0-S_1) \ ; \ K_{32} = -(\gamma_{32}S_0+1)/(S_0-S_1)$$

Now, let $X_i(t)$ = number of type i jobs, $i = 1,2$, processing at work time t (tagged job not included). Suppose the tagged job is of type I. Put

$$Y_i(t) = (X_i(t)-Mm_i)/M. \qquad (5.23)$$

Then, $Y_i(t)$ satisfies the following stochastic differential equations

$$dY_1(t) = -a_1Y_1(t)+(-a_2)Y_2(t)+\sigma_1 dW_1(t), \qquad (5.24)$$

$$dY_2(t) = -b_1Y_1(t)-(-b_2)Y_2(t)+\sigma_2 dW_2(t), \qquad (5.25)$$

where $W_1(t)$ and $W_2(t)$ are independent Brownian motions.

The response time for the tagged job requiring t units of processing time is

$$R(t) = \int_0^t [X_1(u) + X_2(u)]du \tag{5.26}$$

$$= \int_0^t [\sqrt{M}\, Y_1(u) + Mm_1 + \sqrt{M}\, Y_2(u) + Mm_2]du$$

$$= [M(m_1+m_2)]t + \sqrt{M}\int_0^t [Y_1(u)+Y_2(u)]du.$$

Let $Z(t) = \int_0^t [Y_1(u)+Y_2(u)]du$. Then $Z(t)$ satisfies the stochastic differential equation

$$dZ(t) = Y_1(t)+Y_2(t). \tag{5.27}$$

The heavy traffic approximation for the distribution of $R(t)$ is that $R(t)$ has a normal distribution with mean

$$[M(m_1+m_2)]t$$

and variance equal to $M(\mathrm{Var}[Z(t)])$.

To derive an expression for $\mathrm{Var}[Z(t)]$ we will apply results found in Arnold [Ref. 13] to solve the following system of stochastic differential equations:

$$d\underset{\sim}{V}(t) = A\underset{\sim}{V}(t) + Bd\underset{\sim}{W}(t), \tag{5.28}$$

where $\underset{\sim}{V}(t) = [Y_1(t), Y_2(t), Z(t)]$, $d\underset{\sim}{W}(t) = [dW_1(t), dW_2(t)]$

and

$$A = \begin{bmatrix} -a_1 & -a_2 & 0 \\ -b_1 & -b_2 & 0 \\ 1 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \\ 0 & 0 \end{bmatrix}$$

By corollary (8.2.4) on page 130, Arnold states the solution of equation (5.28) as follows.

$$\begin{aligned} Y_1(t) &= [\beta_{11} e^{s_0 t} + K_{11} e^{s_1 t}]Y_1(0) + [\beta_{12} e^{s_0 t} + K_{12} e^{s_1 t}]Y_2(0) \\ &\quad + \int_0^t \sigma_1[\beta_{11} e^{s_0(t-u)} + K_{11} e^{s_1(t-u)}]dW_1(u) \\ &\quad + \int_0^t \sigma_2[\beta_{12} e^{s_0(t-u)} + K_{12} e^{s_1(t-u)}]dW_2(u), \end{aligned} \tag{5.29}$$

$$\begin{aligned} Y_2(t) &= [\beta_{21} e^{s_0 t} + K_{21} e^{s_1 t}]Y_1(0) + [\beta_{22} e^{s_0 t} + K_{22} e^{s_1 t}]Y_2(0) \\ &\quad + \int_0^t \sigma_1[\beta_{21} e^{s_0(t-u)} + K_{21} e^{s_1(t-u)}]dW_1(u) \\ &\quad + \int_0^t \sigma_2[\beta_{22} e^{s_0(t-u)} + K_{22} e^{s_1(t-u)}]dW_2(u), \end{aligned} \tag{5.30}$$

$$\begin{aligned} Z(t) &= [\gamma_{31} + \beta_{31} e^{s_0 t} + K_{31} e^{s_1 t}]Y_1(0) \\ &\quad + [\gamma_{32} + \beta_{32} e^{s_0 t} + K_{32} e^{s_1 t}]Y_2(0) \\ &\quad + \int_0^t \sigma_1[\gamma_{31} + \beta_{31} e^{s_0(t-u)} + K_{31} e^{s_1(t-u)}]dW_1(u) \\ &\quad + \int_0^t \sigma_2[\gamma_{32} + \beta_{32} e^{s_0(t-u)} + K_{32} e^{s_1(t-u)}]dW_2(u). \end{aligned} \tag{5.31}$$

Let $C_1 = \gamma_{31} + \beta_{31} e^{s_0 t} + K_{31} e^{s_1 t}$ and $C_2 = \gamma_{32} + \beta_{32} e^{s_0 t} + K_{32} e^{s_1 t}$. It then follows that

$$
\begin{aligned}
\mathrm{Var}[Z(t)] = {} & C_1^2 \mathrm{Var}[Y_1(0)] + C_2^2 \mathrm{Var}[Y_2(0)] \\
& + 2C_1 C_2 (\mathrm{Cov}[Y_1(0), Y_2(0)]) \qquad (5.32) \\
& + \int_0^t \sigma_1^2 [\gamma_{31} + \beta_{31} e^{s_0(t-u)} + K_{31} e^{s_1(t-u)}]^2 du \\
& + \int_0^t \sigma_2^2 [\gamma_{32} + \beta_{32} e^{s_0(t-u)} + K_{32} e^{s_1(t-u)}]^2 du.
\end{aligned}
$$

where : $\mathrm{Var}[Y_1(0)] = \sigma_1^2 \int_0^\infty [\beta_{11} e^{s_0 u} + K_{11} e^{s_1 u}]^2 du + \sigma_2^2 \int_0^\infty [\beta_{12} e^{s_0 u} + K_{12} e^{s_1 u}]^2 du$

$$\mathrm{Var}[Y_2(0)] = \sigma_1^2 \int_0^\infty [\beta_{21} e^{s_0 u} + K_{21} e^{s_1 u}]^2 du + \sigma_2^2 \int_0^\infty [\beta_{22} e^{s_0 u} + K_{22} e^{s_1 u}]^2 du$$

$$
\begin{aligned}
& \mathrm{Cov}[Y_1(0), Y_2(0)] \\
& = \sigma_1^2 \int_0^\infty [\beta_{11} e^{s_0 u} + K_{11} e^{s_1 u}][\beta_{21} e^{s_0 u} + K_{21} e^{s_1 u}] du \\
& + \sigma_2^2 \int_0^\infty [\beta_{12} e^{s_0 u} + K_{12} e^{s_1 u}][\beta_{22} e^{s_0 u} + K_{22} e^{s_1 u}] du
\end{aligned}
$$

Substitute the integral forms for $Var[Y_1(0)]$, $Var[Y_2(0)]$ and $Cov[Y_1(0),Y_2(0)]$ in equation (5.32). It remains now to solve the simple integrals to obtain a formula for the variance of the response time by heavy traffic approximation.

E. SIMULATION

1. Algorithm

Practically, the simulation for response time in a system with M type I terminals and N type II terminals is performed by applying the same procedure as for the one-type model. This means that, for each initial system state of a tagged job of each type, we generate the exponential arrival and departure times to be able to determine the next system state after a sojourn time. The simulation is for the work time process.

If the minimum of the two exponential times is greater than or equal to the amount of work-time requirement, we simply determine the response time by converting it into real time term. If not, we determine the next system state, the work-time remaining to be accomplished and the accumulated clock time, and generate new exponential times. We repeat this procedure until we obtain a response time for each initial system state of a tagged job of each type. Once we obtain those conditional response times, we can easily calculate the mean, moments, and measures of skewness and kurtosis by applying the steady-state distribution derived previously.

We now describe in detail how to simulate a conditional response time, given an initial system state is (i,j) and the tagged job is of type I, for a computer system consisting of M type I terminals having λ_1 and μ_1 as arrival and requested work rates, and N type II terminals having λ_2 and μ_2 as arrival and requested work rates respectively ; (i does not include the tagged job).

In units of work-time the arrival rate of the next tagged job of either type is

$$A = [(M-(i+1))\lambda_1 + (N-j)\lambda_2](i+j+1), \tag{5.33}$$

and the service completion rate of a job in system is

$$D = [i\mu_1 + j\mu_2]. \tag{5.34}$$

We, then, generate two exponential times with parameters A and D. Call them t' and t" respectively.

If $\min(t',t") \geq w_0$, then set the response time

$$R(i,j,I) = c_0 + w_0(i+j+1),$$

where c_0 is the accumulated clock time (initially = 0) and w_0 is the amount of work-time remains to be completed (initially = T work time requirement of the job).

If $\min(t',t") < w_0$, set

$$w_0 = w_0 - \min(t',t")$$

$$c_0 = c_0 + \min(t',t")(i+j+1).$$

If $\min(t',t") = t'$, which means the system state is changed by an arrival, we generate two more exponential times with parameters $(M-(i+1))\lambda_1(i+j+1)$ and $(N-j)\lambda_2(i+j+1)$ respectively. If the latter quantum of time is less than the previous one, i.e. an arrival of type II job occurs first, the system state changes from (i+1,j) to (i+1,j+1). Otherwise, it changes from (i+1,j) to (i+2,j). In the contrary, if $\min(t',t") = t"$, which means the system state is changed by a departure, we also generate two exponential times for service time with parameters $i\mu_1$ and $j\mu_2$

respectively. If the latter is the minimum of the two, i.e. a departure of type II job occurs first, the system state will then change from (i, j) to (i,j-1). Otherwise, it changes from (i,j) to (i-1,j).

Based on new system state and new work-time requirement, we repeat the same procedure until we obtain the corresponding response time.

Note that we will have in total M times N+1 response times for type I tagged job and M+1 times N for type II tagged job. We can then apply the steady-state distributions $q_I(i,j)$ and $q_{II}(i,j)$ to compute the mean, variance, skewness and kurtosis of the response time of a tagged job that requires T units of processing time as for the one type model.

2. Computation of Simulated Mean, Variance, Skewness and Kurtosis

a. Simulated Mean Response Time

Let $R_k(i,j,I)$ and $R_k(i,j,II)$ be the k-th simulated conditional response time, given the initial system state (tagged job included) is (i, j) and the tagged job is of type I and type II respectively.

If we perform K replications for simulation of response time at each initial system state, then the mean response time, given that the tagged job is of type I, is

$$\bar{R} = \sum_{i=0}^{M} \sum_{j=0}^{N} \bar{R}(i,j,I)\, q_I(i,j), \qquad (5.35)$$

where $\bar{R}(i,j,I) = (\sum_{k=1}^{K} R_k(i,j,I))/K$.

b. Standard Error of the Mean Response Time

Under an assumption that $R_k(i,j,I)$ and $R_k(i,j,II)$ are independent, we can compute the variance of the estimated mean response time, given the tagged job is of type I, based on the value obtained from equation (5.35) as follows.

$$\mathrm{Var}(\bar{R}) = \frac{1}{K}\sum_{i=0}^{M}\sum_{j=0}^{N}\mathrm{Var}[\bar{R}(i,j,I)]q_I^2(i,j) \qquad (5.36)$$

where $\mathrm{Var}[\bar{R}(i,j,I)] = [\sum_{k=1}^{K}(R_k(i,j,I)-\bar{R}(i,j,I))^2]/K$.

To obtain the standard error we simply take the square root of the expression (5.36).

c. Variance of Simulated Response Time

We start by computing higher central moments of response time, given the tagged job is of type I, by applying the same method as for one-type model. The second central moment which is also the variance of response time may be computed, based on simulation data, as follows.

$$\mathrm{Var}[R] = \frac{1}{K}\sum_{k=1}^{K}\sum_{i=0}^{M}\sum_{j=0}^{N}[(R_k(i,j,I)-\bar{R})^2 q_I(i,j)] \qquad (5.37)$$

$$= \frac{1}{K}\sum_{k=1}^{K}\sum_{i=0}^{M}\sum_{j=0}^{N}[(R_k(i,j,I)-\bar{R}(i,j,I))^2$$

$$+ (\bar{R}(i,j,I)-\bar{R})^2]q_I(i,j).$$

d. Skewness of response time

The skewness of response time can be computed by the usual formula, i.e.

$$\gamma_3 = \mu_3 / \sigma^3,$$

where μ_3 is the third central moment and σ is the standard deviation of the response time. As for the calculation of the variance,

$$\mu_3 = \frac{1}{K}\sum_{k=1}^{K}\sum_{i=0}^{M}\sum_{j=0}^{N}[(R_k(i,j,I) - \bar{R})^3 q_I(i,j)] \qquad (5.38)$$

$$= \frac{1}{K}\sum_{k=1}^{K}\sum_{i=0}^{M}\sum_{j=0}^{N}[(R_k(i,j,I) - \bar{R}(i,j,I))^3$$

$$+ 3(R_k(i,j,I) - \bar{R}(i,j,I))^2(\bar{R}(i,j,I) - \bar{R})$$

$$+ (\bar{R}(i,j,I) - \bar{R})^3]q_I(i,j).$$

e. Kurtosis of response time

The kurtosis is defined as $\gamma_4 = (\mu_4 / \sigma^4) - 3$, where μ_4 is the fourth central moment and σ is the standard deviation of the response time. As before;

$$\mu_4 = \frac{1}{K}\sum_{k=1}^{K}\sum_{i=0}^{M}\sum_{j=0}^{N}(R_k(i,j,I) - \bar{R})^4 q_I(i,j) \qquad (5.39)$$

and by simple polynomial expansion;

$$\mu_4 = \frac{1}{K}\sum_{k=1}^{K}\sum_{i=0}^{M}\sum_{j=0}^{N}[(R_k(i,j,I)-\bar{R}(i,j,I))^4$$

$$+ 4(R_k(i,j,I)-\bar{R}(i,j,I))^3(\bar{R}(i,j,I)-\bar{R})$$

$$+ 6(R_k(i,j,I)-\bar{R}(i,j,I))^2(\bar{R}(i,j,I)-\bar{R})^2$$

$$+ (\bar{R}(i,j,I)-\bar{R})^4]q_I(i,j).$$

3. Numerical Results

Tables X and XI show the values of the means, their standard errors (numbers between parentheses below the means), standard deviations, skewness and kurtosis of the empirical response times for a system consisting of 5 type I terminals and 5 type II terminals. We also attach the values of the mean and the standard deviation computed by the central limit theorem and the limiting diffusion in heavy traffic. Those values are computed for two cases. In the first case, Table X, the arrival and departure rates are $\lambda_1 = 30$, $\mu_1 = 100$ for type I jobs and $\lambda_2 = 20$, $\mu_2 = 50$ for type II jobs. But in the second case, Table XI, those rates are in reverse order, i.e. $\lambda_1 = 20$, $\mu_1 = 50$, $\lambda_2 = 30$ and $\mu_2 = 100$.

We see that the results shown in tables X and XI indicate that in both cases the distribution of the empirical response time of a tagged job that requires a fixed amount of processing time is about the same. This indicates that the distribution of the response time is almost independent of the job type of the tagged job. The measures

TABLE X

Simulation results for a two-type model

$M = 5,\ N = 5,\ \lambda_1 = 30,\ \lambda_2 = 20,\ \mu_1 = 100,\ \mu_2 = 50$

Time	Mean	Std.Dev.	Skewness	Kurtosis
0.0100	0.07102	0.01365	-0.80425	-0.77956
	(.00014)			
CLT	0.07156	0.02038		
HTA	0.07101	0.01315		
0.0250	0.17770	0.02676	-0.87139	-0.73673
	(.00032)			
CLT	0.17890	0.03222		
HTA	0.17752	0.02528		
0.0375	0.26720	0.03506	-0.93571	-0.70581
	(.00043)			
CLT	0.26835	0.03947		
HTA	0.26628	0.03249		
0.0500	0.35671	0.04185	-0.98596	-0.70335
	(.00054)			
CLT	0.35780	0.04557		
HTA	0.35503	0.03840		
0.0625	0.44607	0.04758	-0.82744	-0.67426
	(.00062)			
CLT	0.44725	0.05095		
HTA	0.44379	0.04352		

TABLE XI

Simulation Results for two-type model

$M = 5,\ N = 5,\ \lambda_1 = 20,\ \lambda_2 = 30,\ \mu_1 = 50,\ \mu_2 = 100$

Time	Mean	Std.Dev.	Skewness	Kurtosis
0.0100	0.07128	0.01345	-0.79647	-0.76541
	(.00013)			
CLT	0.07065	0.02038		
HTA	0.07101	0.01315		
0.0250	0.17832	0.02632	-0.86195	-0.76107
	(.00031)			
CLT	0.17663	0.03223		
HTA	0.17752	0.02528		
0.0375	0.26638	0.03525	-0.87967	-0.76051
	(.00045)			
CLT	0.26495	0.03947		
HTA	0.26627	0.03249		
0.0500	0.35704	0.04139	-0.86977	-0.75428
	(.00053)			
CLT	0.35327	0.04558		
HTA	0.35503	0.03840		
0.0625	0.44385	0.04777	-0.82988	-0.72768
	(.00063)			
CLT	0.44158	0.05096		
HTA	0.44379	0.04352		

TABLE XII

Approx. Normal VS Empirical quantiles

$M = 5$, $N = 5$, $\lambda_1 = 30$, $\lambda_2 = 20$, $\mu_1 = 100$, $\mu_2 = 50$

Time	Normal	.10	.25	.50	.75	.90	.95	.99
0.0100	CLT	.049	.159	.457	.869	.998	1.0	1.0
	HTA	.114	.229	.442	.718	.919	.979	1.0
0.0250	CLT	.078	.194	.462	.809	.978	.997	1.0
	HTA	.113	.228	.442	.717	.916	.973	.999
0.0375	CLT	.088	.204	.455	.787	.962	.995	1.0
	HTA	.111	.224	.432	.702	.909	.965	.999
0.0500	CLT	.088	.212	.459	.771	.955	.989	1.0
	HTA	.111	.226	.434	.696	.898	.961	.998
0.0625	CLT	.093	.212	.465	.773	.945	.986	1.0
	HTA	.110	.222	.432	.705	.889	.954	.997

$M = 5$, $N = 5$, $\lambda_1 = 20$, $\lambda_2 = 30$, $\mu_1 = 50$, $\mu_2 = 100$

Time	Normal	.10	.25	.50	.75	.90	.95	.99
0.0100	CLT	.041	.137	.433	.838	.998	1.0	1.0
	HTA	.109	.221	.443	.719	.915	.977	1.0
0.0250	CLT	.062	.170	.425	.765	.969	.997	1.0
	HTA	.103	.223	.438	.708	.911	.975	1.0
0.0375	CLT	.079	.191	.429	.754	.952	.991	1.0
	HTA	.118	.229	.447	.713	.905	.969	.999
0.0500	CLT	.075	.181	.417	.724	.939	.986	1.0
	HTA	.106	.222	.431	.693	.897	.965	.998
0.0625	CLT	.081	.198	.436	.745	.932	.980	1.0
	HTA	.115	.238	.451	.719	.898	.960	.997

of skewness for these particular systems indicate that the distribution of the empirical response time is a little skewed to the left, and the measures of kurtosis show thinner tails than it should be for the normal distribution even for a moderately large work-time requirement. We will, therefore, observe the difference between the normal quantiles, approximated by the central limit theorem and by heavy traffic analysis, and the empirical response time to make further judgement on the appropriateness of the approximations.

We now compare the empirical response time at each initial system state to the various quantiles of the normal distributions whose mean and variance are approximated by the central limit theorem and the heavy traffic analysis. The concept of the computations of the normal quantiles and the relative frequencies of the empirical response time is the same as that described for the one-type job model in chapter IV. The results shown in table XII indicate almost the same behavior in distribution of the empirical response time for both cases. Again, this fact indicates that the distribution of the response time is almost independent of the job type of the tagged job. Those values indicate that our normal approximations agree pretty well with the empirical response times obtained by simulation, especially when the work-time requirement becomes large. They also agree with the results shown in tables X and XI where the measures of skewness show a slightly left-skewed distribution and the measures of kurtosis (small negative values) indicate that the distribution of the empirical response times has a slightly flatter peak, fatter shoulders and thinner tails than the normal distribution. Table XII also shows that the empirical response time has a slightly higher mean than that approximated by either central limit theorem or heavy traffic analysis. However, we can still say that the

TABLE XIII

Simulation results for a two-type model

M = 5, N = 5, $\lambda_1 = 40$, $\lambda_2 = 10$, $\mu_1 = 125$, $\mu_2 = 25$

Time	Mean	Std.Dev.	Skewness	Kurtosis
0.0100	0.07246	0.01378	-0.74325	-0.76697
	(.00014)			
CLT	0.07239	0.02491		
HTA	0.07197	0.01328		
0.0250	0.18158	0.02902	-0.88632	-0.76209
	(.00035)			
CLT	0.18096	0.03938		
HTA	0.17993	0.02708		
0.0375	0.27167	0.03886	-0.92108	-0.74667
	(.00051)			
CLT	0.27144	0.04823		
HTA	0.26990	0.03590		
0.0500	0.36227	0.04723	-0.98223	-0.76223
	(.00064)			
CLT	0.36193	0.05569		
HTA	0.35986	0.04326		
0.0625	0.45220	0.05566	-0.94014	-0.74837
	(.00077)			
CLT	0.45241	0.06227		
HTA	0.44983	0.04964		

TABLE XIV

Results for a two-type model

$M = 5,\ N = 5,\ \lambda_1 = 10,\ \lambda_2 = 40,\ \mu_1 = 25,\ \mu_2 = 125$

Time	Std.Dev.	Skewness	Kurtosis	
0.0100	0.07169	0.01335	-0.81156	-0.75757
	(.00014)			
CLT	0.07170	0.02369		
HTA	0.07197	0.01328		
0.0250	0.17877	0.02802	-0.85737	-0.81866
	(.00035)			
CLT	0.17925	0.03746		
HTA	0.17993	0.02708		
0.0375	0.26904	0.03755	-0.92639	-0.82570
	(.00049)			
CLT	0.26888	0.04587		
HTA	0.26990	0.03590		
0.0500	0.35883	0.04561	-1.00062	-0.79135
	(.00062)			
CLT	0.35851	0.05297		
HTA	0.35986	0.04326		
0.0625	0.44810	0.05272	-0.93762	-0.76072
	(.00074)			
CLT	0.44814	0.05922		
HTA	0.44983	0.04964		

TABLE XV

Approx. Normal VS Empirical quantiles

$M = 5$, $N = 5$, $\lambda_1 = 40$, $\lambda_2 = 10$, $\mu_1 = 125$, $\mu_2 = 25$

Time	Normal	.10	.25	.50	.75	.90	.95	.99
0.0100	CLT	.024	.116	.445	.907	1.0	1.0	1.0
	HTA	.110	.228	.433	.710	.905	.973	1.0
0.0250	CLT	.054	.164	.435	.820	.993	1.0	1.0
	HTA	.112	.219	.422	.682	.901	.967	1.0
0.0375	CLT	.070	.186	.428	.795	.981	.999	1.0
	HTA	.113	.224	.414	.693	.899	.967	.999
0.0500	CLT	.075	.182	.439	.784	.972	.998	1.0
	HTA	.110	.216	.422	.689	.896	.969	1.0
0.0625	CLT	.083	.200	.443	.765	.959	.997	1.0
	HTA	.118	.222	.427	.683	.891	.958	1.0

$M = 5$, $N = 5$, $\lambda_1 = 10$, $\lambda_2 = 40$, $\mu_1 = 25$, $\mu_2 = 125$

Time	Normal	.10	.25	.50	.75	.90	.95	.99
0.0100	CLT	.027	.119	.447	.906	1.0	1.0	1.0
	HTA	.110	.230	.458	.737	.931	.982	1.0
0.0250	CLT	.061	.178	.451	.824	.991	.999	1.0
	HTA	.121	.236	.464	.734	.930	.985	1.0
0.0375	CLT	.068	.191	.441	.790	.979	.999	1.0
	HTA	.117	.239	.451	.725	.927	.983	1.0
0.0500	CLT	.081	.189	.430	.776	.975	.997	1.0
	HTA	.117	.232	.444	.729	.935	.985	1.0
0.0625	CLT	.085	.199	.442	.772	.960	.995	1.0
	HTA	.121	.237	.456	.737	.923	.979	1.0

distribution of the empirical response time in table XII is not far from being normal especially when the work requirement is large. The heavy traffic analysis gives a better approximation for small amount of work requirement, while for large work requirement the central limit theorem shows a slightly better approximation.

More results are shown in tables XIII and XIV for the case of $\lambda_1 = 40$, $\lambda_2 = 10$, $\mu_1 = 125$, $\mu_2 = 25$ and the case of reverse rates respectively. The values in both tables do not differ much. This again indicates that the distribution of the response time is independent of the job type of the initial tagged job. The comparison of the empirical response time to the quantiles approximated by the central limit theorem and the limiting diffusion in heavy traffic shown in table XV also suggests the same conclusion. Finally, tables XVI and XVII show the results for a single job type model with the arrival and service rates from the average rates of the two job type model, i.e. $N = 10$, $\lambda = 25$, $\mu = 75$. The values in those tables suggest that there might be some particular cases where we can use the single job type model to approximate the two job type model. However, further investigation is not included in this thesis.

TABLE XVI

Simulation results for one-type model

$N = 10, \lambda = 25, \mu = 75$

Time	Mean	Std.Dev.	Skewness	Kurtosis
0.0100	0.07008	0.01391	-0.70894	0.70894
	(.00024)			
CLT	0.07008	0.02058		
ETA II	0.07000	0.01345		
0.0250	0.17658	0.02629	-0.94906	1.66179
	(.00053)			
CLT	0.17520	0.03253		
ETA II	0.17500	0.02576		
0.0375	0.26202	0.03607	-0.85598	1.32022
	(.00078)			
CLT	0.26280	0.03984		
ETA II	0.26250	0.03302		
0.0500	0.35014	0.04299	-0.76443	1.08030
	(.00094)			
CLT	0.35041	0.04601		
ETA II	0.35000	0.03897		
0.0625	0.43879	0.04783	-0.81254	1.31538
	(.00106)			
CLT	0.43801	0.05144		
ETA II	0.43750	0.04412		

TABLE XVII

Quantile comparison for one-job model

N = 10, λ = 25, μ = 75

Time	Normal	.10	.25	.50	.75	.90	.95	.99
0.0100	CLT	.044	.157	.445	.849	.997	1.0	1.0
	HTA I	.060	.176	.443	.814	.988	1.0	1.0
	HTA II	.117	.235	.443	.722	.912	.976	1.0
0.0250	CLT	.067	.161	.424	.789	.972	.999	1.0
	HTA I	.080	.182	.418	.754	.946	.991	1.0
	HTA II	.093	.204	.418	.712	.912	.973	1.0
0.0375	CLT	.089	.203	.464	.764	.957	.994	1.0
	HTA I	.103	.221	.460	.740	.934	.984	.999
	HTA II	.114	.235	.460	.719	.907	.970	.999
0.0500	CLT	.092	.214	.447	.763	.947	.985	.999
	HTA I	.106	.234	.445	.734	.916	.968	.998
	HTA II	.116	.243	.445	.715	.901	.959	.997
0.0625	CLT	.083	.209	.440	.756	.940	.985	.999
	HTA I	.102	.223	.437	.717	.914	.975	.999
	HTA II	.111	.233	.437	.707	.899	.963	.998

VI. SUMMARY AND CONCLUSION

A. SUMMARY

We brought up a concept of processor-sharing and then described a model for processor-shared work-time allocation of a computer system with one type of terminals as a birth-death process under Markov assumptions in the first chapter. In the second chapter, the mean response time of a tagged job that requires a fixed amount of processing time was derived based on a condition that the tagged job enters to find an initial number of jobs present for processing. We also derived some higher moments of the response time by means of differential equations which led to the computations of variance, skewness and kurtosis of the distribution. Some numerical results obtained by solving those differential equations were then given as an example for a system with two terminals and the arrival and departure rates are 1 and 2 respectively.

A simulation procedure for the response time, given an initial system state when the job was tagged to the system, was described in Chapter III. We detailed an algorithm for a particular two-terminal system to check with the numerical method and generalized it for a computer system with a fixed number of terminals. Based on those empirical response times we showed how to compute the mean response time using the steady-state distribution. The standard errors of the mean response time, the empirical moments and the measures of skewness and kurtosis of the response time were also derived in that chapter. In Chapter IV, we derived the formulas to compute the approximate normal mean and variance of the response time by the central limit theorem for additive

functionals of a birth-death process and also by the limiting diffusion approximation in heavy traffic situation. Based on those approximated means and variances, we explained how to compute the various quantiles and compared them to the empirical response times obtained by simulation. The difference between the relative frequencies of the empirical data and the CDF of the approximated normal distributions indicates how well the normal distributions agree with the empirical response times.

In Chapter V, we described a model for a computer system with two types of terminals, again under Markov assumptions for a bivariate birth and death process. Only now we have to consider a condition that the tagged job is one of the two types. The steady-state distribution was then derived for this two-type system. This allows us to remove the condition of initial system state and leads to the computations of the mean response time of a tagged job that requires a fixed amount of processing time as for one-type model. The computations of higher moments, thus the measures of skewness and kurtosis, were also explained as well as the standard error of the mean. Two approximations for a normal distribution, one based on the central limit theorem for additive functionals and the other one based on the limiting diffusion in heavy traffic situation, were derived for this two-type model in the same way as for the one-type model. Next, a procedure to simulate the response time of a tagged job given an initial system state was described, and then based on those empirical data we make comparison to the quantiles of the approximating normal distributions to study the behavior of the empirical response times towards normality.

Finally, for appendixes, we attached two programs written in FORTRAN to perform the simulation of the response times as described in Chapters III and V respectively. The program in appendix A simulates the response time of a

tagged job based on its work-time requirement, the size of the system and the arrival and departure rates for a single job type model. It computes the statistic elements of the empirical response times such as mean, variance, skewness and kurtosis. It also computes the quantiles of the normal distributions whose mean and variance are approximated by the central limit theorem and heavy traffic analysis. It then computes the relative frequencies of the empirical response times compared to the quantiles. The standard error of each relative frequency is also computed. This will help us to make a judgement on the number of replications needed for simulation. The program in appendix B provides the same things for a two-type job model. We only have to input the mean and variance for the approximation by the central limit theorem from an APL program due to a large number of matrix operations.

B. CONCLUSION

The agreement of the normal approximations with the simulation is satisfactory for both one-type and two-type models, especially in the cases of large work-time requirement and/or under heavy traffic situations. The central limit theorem approximation always works when the work requirement is large enough even if we don't have a heavy traffic situation. The approximation based on the heavy traffic analysis seems, however, to work better, given that the system is under a heavy traffic situation, than the central limit theorem approach for small work requirement. But in a light traffic situation, the limiting diffusion approximation does not work well at all.

Based on the values of skewness and kurtosis of the empirical response time, we have tried to use the Edgeworth expansion to improve the normal distribution of the response

time approximated by the central limit theorem. The distribution computed by the Edgeworth approximation is close to the CLT normal distribution, and the results do not indicate significant improvement.

In the two-type model, we observe that the response time distribution appears to be almost independent of the type of the tagged job. It is hard to make a judgement on the behavior of the empirical response time based on the observations of the mean, variance, skewness and kurtosis, even though those values indicate that the distribution is well approximated by the normal. The simulation for the response time of the two-type model involves a lot of computations. For example, in a system consisting of 5 type I and 5 type II terminals, we have to consider 30 initial system states in total. If we want to perform a simulation with 500 replications, it will involve at least 15,000 computations. This fact indicates that the round-off error may be substantial. However, the relative frequencies of the empirical response time comparing to the approximate normal quantiles obtained for the particular cases in this thesis are in a rather good agreement with the theoretical suggestion. This means that, under the Markov assumptions, the distribution of the response time of a tagged job that requires a fixed (large) amount of processing time is approximately normal.

Throughout this thesis we have been assuming that the distributions of the arrival and service times are exponential. However, all the computations done for both models can as well be extended to the case of general distribution of service time. It might be of interest to study the models and derive a method to simulate the response time, since in the real world we would always have to deal with a more and more sophisticated computer system which is capable of dealing with many kinds of work. So further study might be needed for such system.

APPENDIX A

SIMULATION PROGRAM FOR ONE-TYPE MODEL

```
C*****PROGRAM TO SIMULATE RESPONSE TIME IN PROCESSOR SHARING            PSN00010
C*****OF N TERMINALS COMPARING TO NORMAL DISTRIBUTION.                  PSN00020
C                                                                       PSN00030
C VARIABLES AND CONSTANTS                                               PSN00040
C     N = NUMBER OF TERMINALS                                           PSN00050
C     T = INITIALLY REQUIRED AMOUNT OF WORK                             PSN00060
C     WO = AMT. OF WORK REQUIRED IN FUTURE                              PSN00070
C     CO = AMT. OF CLOCK TIME ACCUMULATED                               PSN00080
C     LAMDA = RATE AT WHICH A JOB APPLIES FOR COMPUTER SERVICE          PSN00090
C     MU = RATE AT WHICH WAITING JOBS ARE SERVED                        PSN00100
C     RJ = FRACTION OF TIME THE PROCESSOR ACTUALLY SPENDS               PSN00110
C          PROCESSING WHEN THERE ARE J JOBS BEING PROCESSED             PSN00120
C     RTJ = RESPONSE TIME WHEN J JOBS BEING PROCESSED                   PSN00130
C     LAMJ = RATE AT WHICH A JOB APPLIES FOR SERVICE WHEN J JOBS        PSN00140
C            ARE IN SYSTEM                                              PSN00150
C     MUJ = DEPARTURE RATE WHEN J JOBS ARE IN SYSTEM                    PSN00160
C                                                                       PSN00170
C DECLARATIONS                                                          PSN00180
      REAL T,LAMDA,MU,RJ(50),RTJ(50,1000),QJ(50),PSI,SMSQ               PSN00190
     *,Q0,QQ(50),LAMQJ(50),MUQJ(50),MCLT,VCLT,SDLT                      PSN00200
     *,VAR,SKEW,KURT,LAMJ(50),MUJ(50),Q(7),QT(7),FX(7)                  PSN00210
     *,PRJ(10,7),PQT(7),PR(7),ALPHA,BETSQ,RO,F,VARP(7),SDP(7)           PSN00220
     *,ER1(50),ER2(50),ER3(50),ER4(50),E1R,E2R,E3R,E4R                  PSN00230
     *,SER,SKN,KTS,DEV(50),D4(50),PI,DM1(7),DM2(7),DM3(7)               PSN00240
     *,BIAS,SKY,KTY,SDR,CDF(5,7),CF(7,5)                                PSN00250
      INTEGER N,KO(7),KT                                                PSN00260
C                                                                       PSN00270
      PI = 3.1415927                                                    PSN00280
      Q(1) = -1.2816                                                    PSN00290
      Q(2) = -0.6745                                                    PSN00300
      Q(3) = 0.                                                         PSN00310
      Q(4) = 0.6745                                                     PSN00320
      Q(5) = 1.2816                                                     PSN00330
      Q(6) = 1.6449                                                     PSN00340
      Q(7) = 2.3263                                                     PSN00350
C ENTER SUBROUTINE VARIABLES                                            PSN00360
      READ(5,500) KT,N,T,LAMDA,MU                                       PSN00370
C                                                                       PSN00380
C READ FRACTION OF TIME PROCESSOR SPENDS PROCESSING                     PSN00390
      DO 5 II=1,N                                                       PSN00400
        RJ(II) = 1.                                                     PSN00410
    5 CONTINUE                                                          PSN00420
```

```
C                                                                       PSN00430
C COMPUTE LAMJ(J) AND MUJ(J)                                            PSN00440
      DO 10 JJ=1,N                                                      PSN00450
        LAMJ(JJ) = LAMDA*(N-JJ)                                         PSN00460
        MUJ(JJ) = MU*(JJ-1)*RJ(JJ)/JJ                                   PSN00470
   10 CONTINUE                                                          PSN00480
C                                                                       PSN00490
C COMPUTE PROB. OF BEING IN STATE J,I.E. QJ                             PSN00500
      DO 13 KK=1,N                                                      PSN00510
        LAMQJ(KK) = LAMDA*(N+1-KK)                                      PSN00520
        MUQJ(KK) = MU*RJ(KK)                                            PSN00530
   13 CONTINUE                                                          PSN00540
      DO 15 KJ=1,N                                                      PSN00550
        QQ(KJ) = 1.                                                     PSN00560
        DO 14 KL=1,KJ                                                   PSN00570
          QQ(KJ) = QQ(KJ)*LAMQJ(KL)/MUQJ(KL)                            PSN00580
   14   CONTINUE                                                        PSN00590
   15 CONTINUE                                                          PSN00600
      QO = 0.                                                           PSN00610
      DO 17 JK=1,N                                                      PSN00620
        QO = QO+QQ(JK)                                                  PSN00630
   17 CONTINUE                                                          PSN00640
      DO 18 IK=1,N                                                      PSN00650
        QJ(IK) = QQ(IK)/QO                                              PSN00660
   18 CONTINUE                                                          PSN00670
C                                                                       PSN00680
C CALL SUBROUTINE SIMULATING RESPONSE TIME                              PSN00690
      CALL SIMPSN(KT,N,T,LAMJ,MUJ,RJ,RTJ)                               PSN00700
      WRITE(6,600) N,KT,T,LAMDA,MU                                      PSN00710
      DO 990 J=1,N                                                      PSN00720
        ER1(J) = 0.                                                     PSN00730
        ER2(J) = 0.                                                     PSN00740
        ER3(J) = 0.                                                     PSN00750
        ER4(J) = 0.                                                     PSN00760
  990 CONTINUE                                                          PSN00770
      DO 1000 J=1,N                                                     PSN00780
        DO 999 K=1,KT                                                   PSN00790
          ER1(J) = ER1(J)+RTJ(J,K)/FLOAT(KT)                            PSN00800
  999   CONTINUE                                                        PSN00810
 1000 CONTINUE                                                          PSN00820
      E1R = 0.                                                          PSN00830
      DO 1010 J=1,N                                                     PSN00840
        E1R = E1R+ER1(J)*QJ(J)                                          PSN00850
 1010 CONTINUE                                                          PSN00860
      DO 1020 J=1,N                                                     PSN00870
        DO 1019 K=1,KT                                                  PSN00880
          ER2(J) = ER2(J)+(RTJ(J,K)-ER1(J))**2/FLOAT(KT)                PSN00890
          ER3(J) = ER3(J)+(RTJ(J,K)-ER1(J))**3/FLOAT(KT)                PSN00900
```

```
          ER4(J) = ER4(J)+(RTJ(J,K)-ER1(J))**4/FLOAT(KT)               PSN00910
 1019   CONTINUE                                                          PSN00920
 1020 CONTINUE                                                            PSN00930
      E2R = 0.                                                            PSN00940
      E3R = 0.                                                            PSN00950
      E4R = 0.                                                            PSN00960
      SER = 0.                                                            PSN00970
      DO 1030 J=1,N                                                       PSN00980
        DEV(J) = ER1(J)-E1R                                               PSN00990
        E2R  = E2R+(ER2(J)+DEV(J)**2)*QJ(J)                               PSN01000
        E3R  = E3R+(ER3(J)+3*ER2(J)*DEV(J)+DEV(J)**3)*QJ(J)               PSN01010
        D4(J) = ER4(J)+4*ER3(J)*DEV(J)+6*ER2(J)*DEV(J)**2+DEV(J)**4       PSN01020
        E4R  = E4R+D4(J)*QJ(J)                                            PSN01030
        SER  = SER+ER2(J)*QJ(J)**2/FLOAT(KT)                              PSN01040
 1030 CONTINUE                                                            PSN01050
      SDR = E2R**0.5                                                      PSN01060
      SKN = E3R/E2R**1.5                                                  PSN01070
      KTS = E4R/E2R**2 - 3                                                PSN01080
      SER = SQRT(SER)                                                     PSN01090
      WRITE(6,663) E1R,SER,SDR,E3R,SKN,E4R,KTS                            PSN01100
C CALL SUBROUTINE TO APPROXIMATE MEAN RESPONSE TIME AND VARIANCE          PSN01110
      CALL CLTRT(N,LAMDA,MU,RJ,PSI,SMSQ)                                  PSN01120
      DO 70 LM=1,5                                                        PSN01130
       IF( (LM .EQ. 1) .OR. (LM .EQ. 2) ) GO TO 300                       PSN01140
        IF (LM .NE. 3 ) GO TO 150                                         PSN01150
         MCLT = T/PSI                                                     PSN01160
         VCLT = T*SMSQ/PSI**3                                             PSN01170
         WRITE(6,660)                                                     PSN01180
         GO TO 200                                                        PSN01190
  150  IF ( LM .NE. 4 ) GO TO 160                                         PSN01200
C APPROX. BY HEAVY TRAFFIC ANALYSIS I.                                    PSN01210
        WRITE(6,661)                                                      PSN01220
        ALPHA = N-MU/LAMDA                                                PSN01230
        BETSQ = 2*MU/(LAMDA**2*ALPHA)                                     PSN01240
        MCLT = T*ALPHA                                                    PSN01250
        VCLT = T*BETSQ                                                    PSN01260
        GO TO 200                                                         PSN01270
  160 WRITE(6,662)                                                        PSN01280
C APPROX. BY HEAVY TRAFFIC ANALYSIS II.                                   PSN01290
      MCLT = T*(N-MU/LAMDA)                                               PSN01300
      RO = LAMDA*N-MU                                                     PSN01310
      F = 1-(1-EXP(-RO*T))/(RO*T)                                         PSN01320
      VCLT = 2*MU*T*F/(LAMDA*RO)                                          PSN01330
  200 SDLT = VCLT**0.5                                                    PSN01340
      WRITE(6,601) MCLT,SDLT                                              PSN01350
C                                                                         PSN01360
C COMPUTE QUANTILES OF NORMAL DIST. WITH MEAN MCLT AND VARIANCE VCLT      PSN01370
C FOR .10, .25, .50, .75, .90, .95, .99                                   PSN01380
```

```
      DO 210  I=1,7                                                     PSN01390
        QT(I) = MCLT+SDLT*Q(I)                                          PSN01400
  210 CONTINUE                                                          PSN01410
      WRITE(6,602) QT                                                   PSN01420
C                                                                       PSN01430
C     WRITE(6,603)                                                      PSN01440
C                                                                       PSN01450
C COMPUTE RELATIVE FREQUENCIES FROM SIMULATION DATA                     PSN01460
      DO 50  I=1,N                                                      PSN01470
       DO 19 L=1,7                                                      PSN01480
         KO(L) = 0                                                      PSN01490
   19  CONTINUE                                                         PSN01500
        DO 46  J=1,KT                                                   PSN01510
          IF( RTJ(I,J) .GT. QT(1)) GO TO 30                             PSN01520
           DO 100 K=1,7                                                 PSN01530
            KO(K) = KO(K)+1                                             PSN01540
  100      CONTINUE                                                     PSN01550
            GO TO 45                                                    PSN01560
   30     IF( RTJ(I,J) .GT. QT(2)) GO TO 31                             PSN01570
           DO 101 K=2,7                                                 PSN01580
            KO(K) = KO(K)+1                                             PSN01590
  101      CONTINUE                                                     PSN01600
            GO TO 45                                                    PSN01610
   31     IF( RTJ(I,J) .GT. QT(3)) GO TO 32                             PSN01620
           DO 102 K=3,7                                                 PSN01630
            KO(K) = KO(K)+1                                             PSN01640
  102      CONTINUE                                                     PSN01650
            GO TO 45                                                    PSN01660
   32     IF( RTJ(I,J) .GT. QT(4)) GO TO 33                             PSN01670
           DO 103 K=4,7                                                 PSN01680
            KO(K) = KO(K)+1                                             PSN01690
  103      CONTINUE                                                     PSN01700
            GO TO 45                                                    PSN01710
   33     IF( RTJ(I,J) .GT. QT(5)) GO TO 34                             PSN01720
           DO 104 K=5,7                                                 PSN01730
            KO(K) = KO(K)+1                                             PSN01740
  104      CONTINUE                                                     PSN01750
            GO TO 45                                                    PSN01760
   34     IF( RTJ(I,J) .GT. QT(6)) GO TO 35                             PSN01770
           DO 105 K=6,7                                                 PSN01780
            KO(K) = KO(K)+1                                             PSN01790
  105      CONTINUE                                                     PSN01800
            GO TO 45                                                    PSN01810
   35     IF( RTJ(I,J) .GT. QT(7)) GO TO 45                             PSN01820
            KO(7) = KO(7)+1                                             PSN01830
   45     CONTINUE                                                      PSN01840
   46   CONTINUE                                                        PSN01850
C                                                                       PSN01860
```

```
            DO 48 K=1,7                                                 PSN01870
              PRJ(I,K) = FLOAT(KO(K))/(FLOAT(KT))                       PSN01880
C             PR(K) = PRJ(I,K)                                          PSN01890
   48       CONTINUE                                                    PSN01900
C           WRITE(6,604) I,PR                                           PSN01910
   50 CONTINUE                                                          PSN01920
C                                                                       PSN01930
      DO 60 K=1,7                                                       PSN01940
        PQT(K) = 0.                                                     PSN01950
        DO 59 I=1,N                                                     PSN01960
          PQT(K) = PQT(K)+PRJ(I,K)*QJ(I)                                PSN01970
   59   CONTINUE                                                        PSN01980
   60 CONTINUE                                                          PSN01990
      WRITE(6,605) PQT                                                  PSN02000
      DO 62 J=1,7                                                       PSN02010
        CDF(LM,J) = PQT(J)                                              PSN02020
   62 CONTINUE                                                          PSN02030
      DO 65 K=1,7                                                       PSN02040
        VARP(K) = 0.                                                    PSN02050
        DO 64 I=1,N                                                     PSN02060
          VARP(K) = VARP(K)+(PQT(K)*(1-PQT(K))*QJ(I)**2)/(FLOAT(KT))    PSN02070
   64   CONTINUE                                                        PSN02080
        SDP(K) = VARP(K)**0.5                                           PSN02090
   65 CONTINUE                                                          PSN02100
      WRITE(6,606) SDP                                                  PSN02110
      GO TO 70                                                          PSN02120
C APPROX. BY EDGEWORTH EXPANSION                                        PSN02130
  300 IF( LM .NE. 1 ) GO TO 305                                         PSN02140
        SKY = SKN                                                       PSN02150
        KTY = KTS                                                       PSN02160
        WRITE(6,666)                                                    PSN02170
        GO TO 307                                                       PSN02180
  305 MCLT = T/PSI                                                      PSN02190
      VCLT = T*SMSQ/PSI**3                                              PSN02200
      SDLT = VCLT**0.5                                                  PSN02210
      WRITE(6,664) MCLT, SDLT                                           PSN02220
      BIAS = E1R-MCLT                                                   PSN02230
      SKY = SKN+(3*BIAS*E2R+BIAS**3)/E2R**1.5                           PSN02240
      KTY = KTS+(4*BIAS*E3R+6*BIAS**2*E2R+BIAS**4)/E2R**2               PSN02250
  307 DO 310 I=1,7                                                      PSN02260
       DM1(I) = (Q(I)**2-1)*SKY/6.                                      PSN02270
       DM2(I) = (Q(I)**3-3*Q(I))*KTY/24.                                PSN02280
       DM3(I) = (Q(I)**5-10*Q(I)**3+15*Q(I))*SKY**2/72.                 PSN02290
       FX(I) = (DM1(I)+DM2(I)+DM3(I))*EXP(-(Q(I)**2/2.))/(SQRT(2*PI))   PSN02300
  310 CONTINUE                                                          PSN02310
       FX(1) = 0.10 - FX(1)                                             PSN02320
       FX(2) = 0.25 - FX(2)                                             PSN02330
       FX(3) = 0.50 - FX(3)                                             PSN02340
```

```
      FX(4) = 0.75 - FX(4)                                              PSN02350
      FX(5) = 0.90 - FX(5)                                              PSN02360
      FX(6) = 0.95 - FX(6)                                              PSN02370
      FX(7) = 0.99 - FX(7)                                              PSN02380
      WRITE(6,665) FX                                                   PSN02390
      DO 68 J=1,7                                                       PSN02400
        CDF(LM,J) = FX(J)                                               PSN02410
   68 CONTINUE                                                          PSN02420
   70 CONTINUE                                                          PSN02430
      DO 75 I=1,7                                                       PSN02440
        DO 74 J=1,5                                                     PSN02450
          CF(I,J) = CDF(J,I)                                            PSN02460
   74   CONTINUE                                                        PSN02470
   75 CONTINUE                                                          PSN02480
      WRITE(6,670) CF                                                   PSN02490
      STOP                                                              PSN02500
C FORMATS                                                               PSN02510
  500 FORMAT(2I5,3F10.4)                                                PSN02520
  600 FORMAT(//,3X,'NO. OF TERMINALS : ',I4,5X,                         PSN02530
     *'NO. OF SIMULATION REPLICATIONS : ',I6/3X,                        PSN02540
     *'REQUIRED WORK TIME : T = ',F10.4                                 PSN02550
     *//3X,'LAMDA = ',F10.4,10X,'MU = ',F10.4)                          PSN02560
  601 FORMAT(/10X,' MEAN :',F13.6,5X,' STANDARD DEV. :',F13.6)          PSN02570
  602 FORMAT(/3X,' QUANTILES OF NORMAL DISTRIBUTION.'/7X,'.10 Q',5X,    PSN02580
     *'.25 Q',5X,'.50 Q',5X,'.75 Q',5X,'.90 Q',5X,'.95 Q',5X,'.99 Q'    PSN02590
     */3X,7F10.4)                                                       PSN02600
C 603 FORMAT(//3X,' CONDITIONAL RELATIVE FREQUENCIES FROM SIMULATION'   PSN02610
C    */36X,'P .10',5X,'P .25',5X,'P .50',5X,'P .75',5X,'P .90',5X,      PSN02620
C    *'P .95',5X,'P .99'/)                                              PSN02630
C 604 FORMAT(/3X,' # JOBS IN SYSTEM :',I5,5X,7F10.4)                    PSN02640
  605 FORMAT(//3X,' UNCONDITIONAL RELATIVE FREQUENCIES'/20X,'P .10',5X  PSN02650
     *,'P .25',5X,'P .50',5X,'P .75',5X,'P .90',5X,'P .95',5X,'P .99'   PSN02660
     */16X,7F10.4)                                                      PSN02670
  606 FORMAT(/3X,' STD. DEV. : ',7F10.4)                                PSN02680
  660 FORMAT('1'//3X,' NORMAL APPROX. BY CENTRAL LIMIT THEOREM.')       PSN02690
  661 FORMAT('1'//3X,' NORMAL APPROX. BY HEAVY TRAFFIC ANALYSIS I.')    PSN02700
  662 FORMAT('1'//3X,' NORMAL APPROX. BY HEAVY TRAFFIC ANALYSIS II.')   PSN02710
  663 FORMAT(/3X,' SIMULATION MEAN RESPONSE TIME : ',F13.6              PSN02720
     */3X,' STANDARD ERROR OF THE MEAN      : ',F13.6                   PSN02730
     */3X,' STANDARD DEVIATION OF RESPONSE TIME       : ',F13.6         PSN02740
     */3X,' THIRD CENTRAL MOMENT OF RESPONSE TIME     : ',F13.6         PSN02750
     */3X,' SKEWNESS OF RESPONSE TIME                 : ',F13.6         PSN02760
     */3X,' FOURTH CENTRAL MOMENT OF RESPONSE TIME    : ',F13.6         PSN02770
     */3X,' KURTOSIS OF RESPONSE TIME                 : ',F13.6)        PSN02780
  664 FORMAT(///3X,' NORMAL APPROX. BY EDGEWORTH EXPANSION'//3X,        PSN02790
     *' CLT. MEAN : ',F13.6,5X,' CLT. STD. DEV. : ',F13.6)              PSN02800
  665 FORMAT(//3X,' NORMAL  : ',4X,'0.10',6X,'0.25',6X,'0.50',6X,       PSN02810
     *'0.75',6X,'0.90',6X,'0.95',6X,'0.99'//3X,' APPROX. : ',7F10.4)    PSN02820
```

```
  666 FORMAT(///3X,' EDGEWORTH EXPANSION WITH ACTUAL MEAN')                  PSN02830
  670 FORMAT(///3X,' SUMMARY OF NORMAL APPROXIMATIONS'//3X,                  PSN02840
     *' NORMAL CDF  : ',4X,'0.10',6X,'0.25',6X,'0.50',6X,'0.75',6X,          PSN02850
     *'0.90',6X,'0.95',6X,'0.99'//3X,' EW. W/O CLT : ',7F10.4//3X,           PSN02860
     *' EW. W/ CLT.  : ',7F10.4//3X,' CLT APPROX. : ',7F10.4//3X,            PSN02870
     *' HVY TRAF. I  : ',7F10.4//3X,' HVY TRAF. II: ',7F10.4)                PSN02880
      END                                                                    PSN02890
C                                                                            PSN02900
C ***************************************************************            PSN02910
C SUBROUTINE TO SIMULATE RESPONSE TIME                                       PSN02920
C ***************************************************************            PSN02930
      SUBROUTINE SIMPSN(KT,N,T,LAMJ,MUJ,RJ,RTJ)                              PSN02940
      REAL *4 WO,CO,T,LAMJ(50),MUJ(50),RJ(50),RTJ(50,1000),                  PSN02950
     *        EXP1,EXP2,EXPJ                                                 PSN02960
      INTEGER IX1,IX2,N,KT                                                   PSN02970
      IX1 = 182394                                                           PSN02980
      IX2 = 328653                                                           PSN02990
C                                                                            PSN03000
C CONDITIONAL RESPONSE TIME                                                  PSN03010
      DO 100 J=1,N                                                           PSN03020
        DO 99 I=1,KT                                                         PSN03030
          WO = T                                                             PSN03040
          CO = 0.                                                            PSN03050
          K = J                                                              PSN03060
C                                                                            PSN03070
C GENERATE EXP(LAMDA*K*(N-J)/RJ(J))                                          PSN03080
   30     CALL LEXPN(IX1,EXP1,1,1,0)                                         PSN03090
C                                                                            PSN03100
C GENERATE EXP(MU*(J-1))                                                     PSN03110
          CALL LEXPN(IX2,EXP2,1,1,0)                                         PSN03120
C                                                                            PSN03130
          IF (K .NE. 1) GO TO 60                                             PSN03140
            EXP1 = EXP1*RJ(K)/(K*LAMJ(K))                                    PSN03150
            EXPJ = EXP1                                                      PSN03160
            GC TO 80                                                         PSN03170
   60     IF (K .NE. N) GO TO 70                                             PSN03180
            EXP2 = EXP2*RJ(K)/(K*MUJ(K))                                     PSN03190
            EXPJ = EXP2                                                      PSN03200
            GC TO 80                                                         PSN03210
   70     EXP1 = EXP1*RJ(K)/(K*LAMJ(K))                                      PSN03220
          EXP2 = EXP2*RJ(K)/(K*MUJ(K))                                       PSN03230
C                                                                            PSN03240
C GET THE MINIMUM OF EXP1 & EXP2                                             PSN03250
          EXPJ = AMIN1(EXP1,EXP2)                                            PSN03260
   80     IF (EXPJ .LT. WO) GO TO 81                                         PSN03270
            RTJ(J,I) = CO+K*WO/RJ(K)                                         PSN03280
            GC TO 90                                                         PSN03290
   81     CO = CO+EXPJ*K/RJ(K)                                               PSN03300
```

```
          WO = WO-EXPJ                                                  PSNO3310
          IF (EXPJ .EQ. EXP1) GO TO 82                                  PSNO3320
            IF (EXPJ .EQ. EXP2) K=K-1                                   PSNO3330
              GO TO 83                                                  PSNO3340
   82     K = K+1                                                       PSNO3350
   83     IX1 = IX1+1                                                   PSNO3360
          IX2 = IX2+1                                                   PSNO3370
          GO TO 30                                                      PSNO3380
   90   CONTINUE                                                        PSNO3390
   99 CONTINUE                                                          PSNO3400
  100 CONTINUE                                                          PSNO3410
      RETURN                                                            PSNO3420
      END                                                               PSNO3430
C                                                                       PSNO3440
C *****************************************************************     PSNO3450
C SUBROUTINE TO APPROX. RESPONSE TIME BY CENTRAL LIMIT THEOREM          PSNO3460
C *****************************************************************     PSNO3470
      SUBROUTINE CLTRT(N,LAMDA,MU,RJ,PSI,SMSQ)                          PSNO3480
      REAL *4 LAMDA,MU,LAMJP(10),MUJP(10),FJ(10),PP(10),PIO,PIJ(10)     PSNO3490
     *,PSI,NUJ(25),GAMMA,ML(10,10),MPI(10,10),IDN(10,10),MA(10,10)      PSNO3500
     *,MSUM(10,10),MZ(10,10),FMJ(10,1),FMTJ(1,10),SMSQ,SINV(10,10)      PSNO3510
     *,C(10,1),D(10,1),E(1,1),EE,PPI(10),WKAREA(2650),RJ(10)            PSNO3520
     *,IER1,IER2,IER3,IER4                                              PSNO3530
      INTEGER N,NP                                                      PSNO3540
C                                                                       PSNO3550
C COMPUTE F'(J)                                                         PSNO3560
      DO 200 I=1,N                                                      PSNO3570
        FJ(I) = RJ(I)/I                                                 PSNO3580
  200 CONTINUE                                                          PSNO3590
C                                                                       PSNO3600
C COMPUTE LAMCA'(J) AND MU'(J)                                          PSNO3610
      NP = N-1                                                          PSNO3620
      DO 205 J=1,N                                                      PSNO3630
        LAMJP(J) = LAMDA*(N-J)                                          PSNO3640
        MUJP(J) = MU*(J-1)/J                                            PSNO3650
  205 CONTINUE                                                          PSNO3660
C                                                                       PSNO3670
C COMPUTE PI(J)_VECTOR                                                  PSNO3680
      DO 210 II=1,NP                                                    PSNO3690
        PP(II) = 1                                                      PSNO3700
        DO 209 IJ=1,II                                                  PSNO3710
          PP(II) = PP(II)*LAMJP(IJ)/MUJP(IJ+1)                          PSNO3720
  209   CONTINUE                                                        PSNO3730
  210 CONTINUE                                                          PSNO3740
      PIO = 0.                                                          PSNO3750
      DO 212 IK=1,NP                                                    PSNO3760
        PIO = PIO+PP(IK)                                                PSNO3770
  212 CONTINUE                                                          PSNO3780
```

```
      PIO = 1/(1+PIO)                                                   PSN03790
      DO 213 IL=1,NP                                                    PSN03800
         PPI(IL) = PIO*PP(IL)                                           PSN03810
  213 CONTINUE                                                          PSN03820
      DO 220 IM=1,N                                                     PSN03830
         IF (IM .NE. 1) GO TO 215                                       PSN03840
           PIJ(IM) = PIO                                                PSN03850
           GO TO 219                                                    PSN03860
  215    PIJ(IM) = PPI(IM-1)                                            PSN03870
  219    CONTINUE                                                       PSN03880
  220 CONTINUE                                                          PSN03890
C                                                                       PSN03900
C COMPUTE PSI                                                           PSN03910
      PSI = 0.                                                          PSN03920
      DO 230 IN=1,N                                                     PSN03930
         PSI = PSI+FJ(IN)*PIJ(IN)                                       PSN03940
  230 CONTINUE                                                          PSN03950
C                                                                       PSN03960
C COMPUTE SIGMA SQUARE                                                  PSN03970
      DO 250 JI=1,N                                                     PSN03980
         IF (JI .NE. 1) GO TO 240                                       PSN03990
           NUJ(JI) = LAMJP(JI)                                          PSN04000
           GO TO 249                                                    PSN04010
  240    IF (JI .NE. N) GO TO 245                                       PSN04020
           NUJ(JI) = MUJP(JI)                                           PSN04030
           GO TO 249                                                    PSN04040
  245    NUJ(JI) = LAMJP(JI)+MUJP(JI)                                   PSN04050
  249    CONTINUE                                                       PSN04060
  250 CONTINUE                                                          PSN04070
      GAMMA = 0.                                                        PSN04080
      DO 251 KK=1,N                                                     PSN04090
         GAMMA = AMAX1(GAMMA,NUJ(KK))                                   PSN04100
  251 CONTINUE                                                          PSN04110
      DO 270 I=1,N                                                      PSN04120
         DO 260 J=1,N                                                   PSN04130
           ML(I,J) = PIJ(J)                                             PSN04140
           IF (I .NE. J) GO TO 255                                      PSN04150
              MPI(I,J) = PIJ(J)                                         PSN04160
              IDN(I,J) = 1.                                             PSN04170
              MA(I,J) = 1-NUJ(J)/GAMMA                                  PSN04180
              GC TO 260                                                 PSN04190
  255      MPI(I,J) = 0.                                                PSN04200
           IDN(I,J) = 0.                                                PSN04210
           IF (J .NE. (I-1)) GO TO 256                                  PSN04220
              MA(I,J) = MUJP(I)/GAMMA                                   PSN04230
              GC TO 260                                                 PSN04240
  256      IF (J .NE. (I+1)) GO TO 257                                  PSN04250
              MA(I,J) = LAMJP(I)/GAMMA                                  PSN04260
```

```
            GC TO 260                                                   PSN04270
  257     MA(I,J) = 0.                                                  PSN04280
  260   CONTINUE                                                        PSN04290
  270 CONTINUE                                                          PSN04300
C                                                                       PSN04310
      DO 280 K=1,N                                                      PSN04320
        DO 275 L=1,N                                                    PSN04330
          MSUM(K,L) = IDN(K,L)-MA(K,L)+ML(K,L)                          PSN04340
  275   CONTINUE                                                        PSN04350
  280 CONTINUE                                                          PSN04360
C                                                                       PSN04370
      CALL LINV2F(MSUM,N,N,SINV,0,WKAREA,IER1)                          PSN04380
      DO 290 II=1,N                                                     PSN04390
        DO 285 JJ=1,N                                                   PSN04400
          MZ(II,JJ) = (SINV(II,JJ)-ML(II,JJ))/GAMMA                     PSN04410
  285   CONTINUE                                                        PSN04420
  290 CONTINUE                                                          PSN04430
C                                                                       PSN04440
      DO 300 IJ=1,N                                                     PSN04450
        FMJ(IJ,1) = FJ(I J)                                             PSN04460
        FMTJ(1,IJ) = FJ( IJ)                                            PSN04470
  300 CONTINUE                                                          PSN04480
      CALL VMULFF(MZ,FMJ,N,N,1,N,N,C,N,IER2)                            PSN04490
      CALL VMULFF(MPI,C,N,N,1,N,N,D,N,IER3)                             PSN04500
      CALL VMULFF(FMTJ,D,1,N,1,1,N,E,1,IER4)                            PSN04510
C                                                                       PSN04520
      EE = E(1,1)                                                       PSN04530
      SMSQ = 2*EE                                                       PSN04540
      RETURN                                                            PSN04550
      END                                                               PSN04560
```

APPENDIX B

SIMULATION PROGRAM FOR TWO-TYPE MODEL

```
C*****************************************************************
C PROGRAM TO SIMULATE RESPONSE TIME AND COMPARE TC NORMAL APPROX.*
C                FOR TWO-TYPE JOB MODEL                          *
C*****************************************************************
C VARIABLES AND CONSTANTS
C     M = NUMBER OF TYPE 1 TERMINALS
C     N = NUMBER OF TYPE 2 TERMINALS
C     T = INITIALLY REQUIRED AMOUNT OF WORK
C     LAM1 = RATE AT WHICH A JOB ARRIVES AT TYPE 1 TERMINAL
C     LAM2 = RATE AT WHICH A JOB ARRIVES AT TYPE 2 TERMINAL
C     MU1 = RATE AT WHICH WAITING JOBS AT TYPE 1 TERMINAL ARE SERVED
C     MU2 = RATE AT WHICH WAITING JOBS AT TYPE 2 TERMINAL ARE SERVED
C
      REAL T,LAM1,LAM2,MU1,MU2,LJ1(6),LJ2(6),MJ1(6),MJ2(6),SS3
     *,MRT,SE,X1(6,6),V1(6,6),A1(6,6),B1(6,6),QH(7),PHT(7),SKHT
     *,RT3,RT4,Q1,NJ1(6,6),C1(6,6),SS2,SS4,VAR,SDV,SKEW,KURT,KUHT
     *,XBLT,SDLT,VARP(7),SDP(7),PR1(7),PH1(7),VPH(7),HTXB,HTSD
      INTEGER M,N,M1,N1,KT
C
      KT = 300
C ENTER SUBROUTINE VARIABLES
      READ(5,500) M,N,T,LAM1,LAM2,MU1,MU2
      M1 = M+1
      N1 = N+1
C CALL SUBROUTINE COMPUTING STEADY STATE DISTRIBUTION
      CALL SST(M,N,LAM1,LAM2,MU1,MU2,NU1)
C CALL SUBROUTINE SIMULATING CONDITIONAL RESPONSE TIME
      CALL SIMRT(M,N,T,LAM1,LAM2,MU1,MU2,X1,A1,B1,C1
     *,PR1,VARP,PH1,VPH,HTXB,HTSD,QH)
      WRITE(6,601) M,LAM1,MU1,N,LAM2,MU2,T
      MRT = 0.
      DO 270 I=1,M1
        DO 260 J=1,N1
          MRT = MRT+NU1(I,J)*X1(I,J)
  260   CONTINUE
  270 CONTINUE
      DO 272 I=1,M1
        DO 271 J=1,N1
          V1(I,J) = X1(I,J)-MRT
  271   CONTINUE
  272 CONTINUE
      VAR = 0.
```

```
      RT3 =  0.                                                         APP00430
      RT4 =  0.                                                         APP00440
      SE  = 0.                                                          APP00450
      DO  275  I=1,M1                                                   APP00460
       DO 274  J=1,N1                                                   APP00470
        SE=SE+A1(I,J)*NU1(I,J)**2/FLOAT(KT)                             APP00480
        VAR  =  VAR+(A1(I,J)+V1(I,J)**2)*NU1(I,J)                       APP00490
        RT3  =  RT3+(B1(I,J)+3*A1(I,J)*V1(I,J)+V1(I,J)**3)*NU1(I,J)     APP00500
        D5=C1(I,J)+4*B1(I,J)*V1(I,J)+6*A1(I,J)*V1(I,J)**2+V1(I,J)**4    APP00510
        RT4  =  RT4+D5*NU1(I,J)                                         APP00520
  274  CONTINUE                                                         APP00530
  275 CONTINUE                                                          APP00540
      SE  =  SQRT(SE)                                                   APP00550
      SDV  =  SQRT(VAR)                                                 APP00560
      SKEW  =  RT3/VAR**1.5                                             APP00570
      KURT  =  RT4/VAR**2                                               APP00580
      DO  50  K=2,7                                                     APP00590
        QH(K)  =  (QH(K)+QH(K-1))/2.                                    APP00600
        PHT(K)  =  PH1(K)-PH1(K-1)                                      APP00610
   50 CONTINUE                                                          APP00620
      PHT(1)  =  PH1(1)                                                 APP00630
      SS2  =  0.                                                        APP00640
      SS3  =  0.                                                        APP00650
      SS4  =  0.                                                        APP00660
      DO  60  I=1,7                                                     APP00670
        SS2  =  SS2+(QH(I)-HTXB)**2*PHT(I)                              APP00680
        SS3  =  SS3+(QH(I)-HTXB)**2*PHT(I)                              APP00690
        SS4  =  SS4+(QH(I)-HTXB)**4*PHT(I)                              APP00700
   60 CONTINUE                                                          APP00710
      SKHT  =  SS3/SS2**1.5                                             APP00720
      KUHT  =  SS4/SS2**2  -  3.                                        APP00730
      WRITE(6,602) MRT,SE,SDV,RT3,SKEW,RT4,KURT                         APP00740
C CALL SUBROUTINE TO READ CLT MEAN AND VARIANCE                         APP00750
      CALL APPROX(XBLT,SDLT)                                            APP00760
      XBLT  =  XBLT*T                                                   APP00770
      SDLT  =  SQRT(SDLT*T)                                             APP00780
      WRITE(6,603) XBLT,SDLT                                            APP00790
      DO  276  I=1,7                                                    APP00800
        SDP(I)  =  SQRT(VARP(I))                                        APP00810
        VPH(I)  =  SQRT(VPH(I))                                         APP00820
  276 CONTINUE                                                          APP00830
      WRITE(6,604) PR1,SDP                                              APP00840
      WRITE(6,605) HTXB,HTSD,SKHT,KUHT                                  APP00850
      WRITE(6,606) PH1,VPH                                              APP00860
C                                                                       APP00870
      STOP                                                              APP00880
C FORMATS                                                               APP00890
  500 FORMAT(2I5,5F10.4)                                                APP00900
```

```
601 FORMAT('1',//5X,'NO. OF TYPE I TERMINALS  : ',I5,5X,
   *'LAMDA I  = ',F10.4,5X,'MU I  = ',F10.4//5X,
   *'NO. OF TYPE II TERMINALS : ',I5,5X,'LAMDA II = ',F10.4,
   *5X,'MU II = ',F10.4//5X,
   *'REQUIRED WORK TIME : T = ',F13.6)
602 FORMAT(///5X,'MEAN RESPONSE TIME        : ',F13.6//5X,
   *'STD. ERROR OF THE MEAN : ',F13.6//5X,
   *'STANDARD DEVIATION OF RESPONSE TIME    : ',F13.6//5X,
   *'THIRD CENTRAL MOMENT OF RESPONSE TIME  : ',F13.6//5X,
   *'SKEWNESS OF RESPONSE TIME              : ',F13.6//5X,
   *'FOURTH CENTRAL MOMENT OF RESPONSE TIME : ',F13.6//5X,
   *'KURTOSIS OF RESPONSE TIME              : ',F13.6)
603 FORMAT(///3X,' NORMAL APPROXIMATION BY CENTRAL LIMIT THEOREM'
   *//3X,' CLT. MEAN : ',F13.6,5X,' CLT. STD. DEV. : ',F13.6)
604 FORMAT(//3X,' NORMAL CDF  : ',4X,'0.10',6X,'0.25',6X,'0.50'
   *,6X,'0.75',6X,'0.90',6X,'0.95',6X,'0.99'//3X,' CLT APPROX. : '
   *,7F10.4//3X,' STD. ERROR  : ',7F10.4)
605 FORMAT(///3X,' NORMAL APPROXIMATION BY HEAVY TRAFFIC ANALYSIS'
   *,//3X,' HTA. MEAN : ',F13.6,5X,' HTA. STD. DEV. : ',F13.6
   *,//3X,' HTA. SKEWNESS : ',F13.6,5X,' HTA. KURTOSIS : ',F13.6)
606 FORMAT(//3X,' NORMAL CDF  : ',4X,'0.10',6X,'0.25',6X,'0.50'
   *,6X,'0.75',6X,'0.90',6X,'0.95',6X,'0.99'//3X,' HTA APPROX. : '
   *,7F10.4//3X,' STD. ERROR  : ',7F10.4)
    END
C
C*****************************************************************
C SUBROUTINE TO COMPUTE STEADY-STATE DISTRIBUTION FOR TYPE I JOB
C*****************************************************************
    SUBROUTINE SST(M,N,LAM1,LAM2,MU1,MU2,NU1)
    REAL T,LAM1,LAM2,MU1,MU2,NU1(6,6),PP(6,6),PI(6,6),TU1(6,6),Q1
    INTEGER M,N,M1,N1
    M1  = M+1
    N1  = N+1
    DO 100  I=1,M
      DO 90 J=1,N
        FI  = FLOAT(I)
        FJ  = FLOAT(J)
        FM  = FLOAT(M)
        FN  = FLOAT(N)
        PP(I,J) =  GAMMA(FM+1.)*GAMMA(FN+1.)*(LAM1/MU1)**I
        PP(I,J) =  PP(I,J)*(LAM2/MU2)**J/(GAMMA(FI+1.)*GAMMA(FJ+1.))
        PP(I,J) =  PP(I,J)/(GAMMA(FM-FI+1.)*GAMMA(FN-FJ+1.))
        PP(I,J) =  PP(I,J)*GAMMA(FI+FJ+1.)
 90   CONTINUE
100 CONTINUE
    DO 102 I=2,M1
      DO 101 J=2,N1
        PI(I,J) =  PP((I-1),(J-1))
```

```
101     CONTINUE                                                      APP01390
102   CONTINUE                                                        APP01400
      DO 103 J=1,N                                                    APP01410
        PI(1,(J+1))=GAMMA((FLOAT(N+1)))/GAMMA((FLOAT(N-J+1)))         APP01420
        PI(1,(J+1))=PI(1,(J+1))*(LAM2/MU2)**J                         APP01430
103   CONTINUE                                                        APP01440
      DO 104 I=1,M                                                    APP01450
        PI((I+1),1)=GAMMA((FLOAT(M+1)))/GAMMA((FLOAT(M-I+1)))         APP01460
        PI((I+1),1)=PI((I+1),1)*(LAM1/MU1)**I                         APP01470
104   CONTINUE                                                        APP01480
      PI(1,1) = 0.                                                    APP01490
      DO 120 I=1,M1                                                   APP01500
        DO 110 J=1,N1                                                 APP01510
          PI(1,1) = PI(1,1)+PI(I,J)                                   APP01520
110     CONTINUE                                                      APP01530
120   CONTINUE                                                        APP01540
      PI(1,1) = 1./(PI(1,1)+1.)                                       APP01550
      DO 140 I=1,M1                                                   APP01560
        DO 130 J=1,N1                                                 APP01570
         IF ( (I .EQ. 1) .AND. (J .EQ. 1)) GO TO 130                  APP01580
          PI(I,J) = PI(1,1)*PI(I,J)                                   APP01590
130     CONTINUE                                                      APP01600
140   CONTINUE                                                        APP01610
      DO 180 I=1,M1                                                   APP01620
        DO 170 J=1,N1                                                 APP01630
          TU1(I,J) = PI(I,J)*LAM1*(M1-I)                              APP01640
170     CONTINUE                                                      APP01650
180   CONTINUE                                                        APP01660
      Q1 = 0.                                                         APP01670
      DO 200 I=1,M1                                                   APP01680
        DO 190 J=1,N1                                                 APP01690
          Q1 = Q1+TU1(I,J)                                            APP01700
190     CONTINUE                                                      APP01710
200   CONTINUE                                                        APP01720
      Q1 = 1./Q1                                                      APP01730
      DO 220 I=1,M1                                                   APP01740
        DO 210 J=1,N1                                                 APP01750
          TU1(I,J) = TU1(I,J)*Q1                                      APP01760
210     CONTINUE                                                      APP01770
220   CONTINUE                                                        APP01780
      DO 225 J=1,N1                                                   APP01790
        NU1(1,J) = 0.                                                 APP01800
225   CONTINUE                                                        APP01810
      DO 232 I=2,M1                                                   APP01820
        DO 231 J=1,N1                                                 APP01830
          NU1(I,J) = TU1((I-1),J)                                     APP01840
231     CONTINUE                                                      APP01850
232   CONTINUE                                                        APP01860
```

```
  235 CONTINUE                                                          APP01870
      RETURN                                                            APP01880
      END                                                               APP01890
C                                                                       APP01900
C *********************************************************************APP01910
C SUBROUTINE TO SIMULATE RESPONSE TIME AND COMPARISON TO CLT            APP01920
C *********************************************************************APP01930
      SUBROUTINE SIMRT(M,N,T,LAM1,LAM2,MU1,MU2,X1,A1,B1,C1              APP01940
     *,PR1,VARP,PH1,VPH,HTXB,HTSD,QH)                                   APP01950
      REAL *4 W0,C0,T,LAM1,LAM2,MU1,MU2,R1(5,6),XR(1000),AR1(6,6)       APP01960
     *,LJ1(6),LJ2(6),MJ1(6),MJ2(6),EXP(6),XJ,XL,XM,BR1(5,6),CR1(5,6)    APP01970
     *,EXP1,EXP2,EXP3,EXP4,X1(6,6),A1(6,6),B1(6,6),C1(6,6)              APP01980
     *,QU(7),QI(7),PK1(7),PK3(7),PR1(7),PH1(7),VARP(7),DMY(7),QH(7)     APP01990
     *,VPH(7),DNY(7),NU1(6,6),XBLT,SDLT,HTXB,HTSD                       APP02000
      INTEGER IX1,IX2,M,N,M1,N1,KT,K1(7),K3(7)                          APP02010
      IX1 = 253766                                                      APP02020
      IX2 = 344921                                                      APP02030
      M1  = M+1                                                         APP02040
      N1  = N+1                                                         APP02050
      KT  = 300                                                         APP02060
      DO 10 I=1,M1                                                      APP02070
         LJ1(I) = (M1-I)*LAM1                                           APP02080
         MJ1(I) = MU1*(I-1)                                             APP02090
   10 CONTINUE                                                          APP02100
      DO 11 J=1,N1                                                      APP02110
         LJ2(J) = (N1-J)*LAM2                                           APP02120
         MJ2(J) = MU2*(J-1)                                             APP02130
   11 CONTINUE                                                          APP02140
C CALL SUBROUTINE TO COMPUTE STEADY-STATE DISTRIBUTION                  APP02150
      CALL SST(M,N,LAM1,LAM2,MU1,MU2,NU1)                               APP02160
C STD. NORMAL QUANTILES (.10,.25,.50,.75,.90,.95,.99)                   APP02170
      QU(1) = -1.2816                                                   APP02180
      QU(2) = -0.6745                                                   APP02190
      QU(3) = 0.                                                        APP02200
      QU(4) = 0.6745                                                    APP02210
      QU(5) = 1.2816                                                    APP02220
      QU(6) = 1.6449                                                    APP02230
      QU(7) = 2.3263                                                    APP02240
C CALL SUBROUTINE TO READ CLT MEAN AND VARIANCE                         APP02250
      CALL APPROX(XBLT,SDLT)                                            APP02260
      XBLT = XBLT*T                                                     APP02270
      SDLT = SQRT(SDLT*T)                                               APP02280
C CALL SUBROUTINE TO COMPUTE HEAVY TRAFFIC MEAN AND VARIANCE            APP02290
      CALL HTA2(T,M,N,LAM1,LAM2,MU1,MU2,HTXB,HTSD)                      APP02300
      HTSD = SQRT(HTSD)                                                 APP02310
      DO 13 I=1,7                                                       APP02320
         QI(I) = XBLT+SDLT*QU(I)                                        APP02330
         PR1(I) = 0.                                                    APP02340
```

```
        VARP(I) = 0.                                                    APP02350
        QH(I) = HTXB +HTSD*QU(I)                                        APP02360
        PH1(I) = 0.                                                     APP02370
        VPH(I) = 0.                                                     APP02380
   13 CONTINUE                                                          APP02390
C                                                                       APP02400
C CONDITIONAL RESPONSE TIME                                             APP02410
      DO 25 NI=1,M                                                      APP02420
       DO 24 NJ=1,N1                                                    APP02430
        R1(NI,NJ) = 0.                                                  APP02440
        AR1(NI,NJ) = 0.                                                 APP02450
        BR1(NI,NJ) = 0.                                                 APP02460
        CR1(NI,NJ) = 0.                                                 APP02470
   24  CONTINUE                                                         APP02480
   25 CONTINUE                                                          APP02490
      DO 250 I=1,M                                                      APP02500
        DO 240 J=1,N1                                                   APP02510
         DO 230 K=1,KT                                                  APP02520
          W0 = T                                                        APP02530
          C0 = 0.                                                       APP02540
          I0 = I                                                        APP02550
          J0 = J                                                        APP02560
C                                                                       APP02570
C GENERATE EXPONENTIAL ARRIVAL AND DEPARTURE TIME                       APP02580
   30     CALL LEXPN(IX1,EXP,6,1,0)                                     APP02590
C                                                                       APP02600
          IF ( I0 .NE. M ) GO TO 50                                     APP02610
            EXP1 = 9999.99                                              APP02620
            IF ( I0 .NE. 1 ) GO TO 38                                   APP02630
              EXP3 = 9999.99                                            APP02640
              GO TO 39                                                  APP02650
   38       EXP3 = EXP(3)/(MU1*(I0-1))                                  APP02660
   39       IF ( J0 .NE. 1 ) GO TO 40                                   APP02670
              EXP2 = EXP(2)/(LJ2(J0)*(I0+J0-1))                         APP02680
              XL = EXP(5)/((LJ1(I0+1)+LJ2(J0))*(I0+J0-1))               APP02690
              EXP4 = 9999.99                                            APP02700
              XM = EXP3                                                 APP02710
              GO TO 110                                                 APP02720
   40       IF ( J0 .NE. N1 ) GO TO 41                                  APP02730
              EXP2 = 9999.99                                            APP02740
              XL = EXP1                                                 APP02750
              EXP4 = EXP(4)/(MU2*(J0-1))                                APP02760
              XM = EXP(6)/(MU1*(I0-1)+MU2*(J0-1))                       APP02770
            GO TO 110                                                   APP02780
   41       EXP2 = EXP(2)/(LJ2(J0)*(I0+J0-1))                           APP02790
            XL = EXP(5)/((LJ1(I0+1)+LJ2(J0))*(I0+J0-1))                 APP02800
            EXP4 = EXP(4)/(MU2*(J0-1))                                  APP02810
            XM = EXP(6)/(MU1*(I0-1)+MU2*(J0-1))                         APP02820
```

```
          GO TO 110                                                     APP02830
 50     IF ( IO .NE. 1 ) GO TO 55                                       APP02840
          EXP1 = EXP(1)/(LJ1(IO+1)*(IO+JO-1))                           APP02850
          EXP3 = 9999.99                                                APP02860
          GO TO 39                                                      APP02870
 55     EXP1 = EXP(1)/(LJ1(IO+1)*(IO+JO-1))                             APP02880
        EXP3 = EXP(3)/(MU1*(IO-1))                                      APP02890
        GO TO 39                                                        APP02900
 110    XJ = AMIN1(XL,XM)                                               APP02910
        IF ( XJ .LT. WO ) GO TO 120                                     APP02920
          R1(I,J) = R1(I,J)+(CO+WO*(IO+JO-1))/(FLOAT(KT))               APP02930
          XR(K) = CO+WO*(IO+JO-1)                                       APP02940
          GC TO 220                                                     APP02950
 120    CO = CO+XJ*(IO+JO-1)                                            APP02960
        WO = WO-XJ                                                      APP02970
        IF ( XJ .NE. XL ) GO TO 140                                     APP02980
          IF ( EXP1 .GT. EXP2 ) GO TO 130                               APP02990
           IF ( IO .EQ. M ) GO TO 160                                   APP03000
            IO = IO+1                                                   APP03010
            GO TO 160                                                   APP03020
 130      IF ( JO .EQ. N1 ) GO TO 160                                   APP03030
           JO = JO+1                                                    APP03040
           GO TO 160                                                    APP03050
 140    IF ( EXP3 .GT. EXP4 ) GO TO 150                                 APP03060
         IF ( IO .EQ. 1 ) GO TO 160                                     APP03070
          IO = IO-1                                                     APP03080
          GO TO 160                                                     APP03090
 150    IF ( JO .EQ. 1) GO TO 160                                       APP03100
         JO = JO-1                                                      APP03110
 160    IX1 = IX1+1                                                     APP03120
        GO TO 30                                                        APP03130
 220    CONTINUE                                                        APP03140
 230  CONTINUE                                                          APP03150
      DO 231 LI=1,7                                                     APP03160
        K1(LI) = 0                                                      APP03170
        K3(LI) = 0                                                      APP03180
 231  CONTINUE                                                          APP03190
      DO 235 K=1,KT                                                     APP03200
        AR1(I,J) = AR1(I,J)+(XR(K)-R1(I,J))**2/(FLOAT(KT))              APP03210
        BR1(I,J) = BR1(I,J)+(XR(K)-R1(I,J))**3/(FLOAT(KT))              APP03220
        CR1(I,J) = CR1(I,J)+(XR(K)-R1(I,J))**4/(FLOAT(KT))              APP03230
        IF( XR(K) .GT. QI(1) ) GO TO 355                                APP03240
          DO 350 LI=1,7                                                 APP03250
            K1(LI) = K1(LI)+1                                           APP03260
 350      CCNTINUE                                                      APP03270
          GO TO 400                                                     APP03280
 355    IF( XR(K) .GT. QI(2) ) GO TO 360                                APP03290
          DC 357 LI=2,7                                                 APP03300
```

```
            K1(LI) = K1(LI)+1                                           APP03310
357       CCNTINUE                                                      APP03320
          GO TO 400                                                     APP03330
360     IF( XR(K) .GT. QI(3) ) GO TO 365                                APP03340
          DO 363 LI=3,7                                                 APP03350
            K1(LI) = K1(LI)+1                                           APP03360
363       CCNTINUE                                                      APP03370
          GO TO 400                                                     APP03380
365     IF( XR(K) .GT. QI(4) ) GO TO 370                                APP03390
          DC 367 IL=4,7                                                 APP03400
            K1(IL) = K1(IL)+1                                           APP03410
367       CCNTINUE                                                      APP03420
          GO TO 400                                                     APP03430
370     IF( XR(K) .GT. QI(5) ) GO TO 375                                APP03440
          DC 372 KI=5,7                                                 APP03450
            K1(KI) = K1(KI)+1                                           APP03460
372       CCNTINUE                                                      APP03470
          GO TO 400                                                     APP03480
375     IF( XR(K) .GT. QI(6) ) GO TO 380                                APP03490
          DC 377 LK=6,7                                                 APP03500
            K1(LK) = K1(LK)+1                                           APP03510
377       CCNTINUE                                                      APP03520
          GC TO 400                                                     APP03530
380     IF( XR(K) .GT. QI(7) ) GO TO 400                                APP03540
          K1(7) = K1(7)+1                                               APP03550
400     CONTINUE                                                        APP03560
235   CONTINUE                                                          APP03570
      DO 800 MI=1,KT                                                    APP03580
        IF( XR(MI) .GT. QH(1) ) GO TO 710                               APP03590
          DO 705 MK=1,7                                                 APP03600
            K3(MK) = K3(MK)+1                                           APP03610
705       CONTINUE                                                      APP03620
          GC TO 780                                                     APP03630
710     IF( XR(MI) .GT. QH(2) ) GO TO 720                               APP03640
          DC 715 MKK=2,7                                                APP03650
            K3(MKK) = K3(MKK)+1                                         APP03660
715       CCNTINUE                                                      APP03670
          GC TO 780                                                     APP03680
720     IF( XR(MI) .GT. QH(3) ) GO TO 730                               APP03690
          DO 725 MKI=3,7                                                APP03700
            K3(MKI) = K3(MKI)+1                                         APP03710
725       CCNTINUE                                                      APP03720
          GC TO 780                                                     APP03730
730     IF( XR(MI) .GT. QH(4) ) GO TO 740                               APP03740
          DO 735 MKJ=4,7                                                APP03750
            K3(MKJ) = K3(MKJ)+1                                         APP03760
735       CCNTINUE                                                      APP03770
          GO TO 780                                                     APP03780
```

```
740     IF( XR(MI) .GT. QH(5) ) GO TO 750                               APP03790
          DO 745 MKL=5,7                                                APP03800
            K3(MKL) = K3(MKL)+1                                         APP03810
745       CONTINUE                                                      APP03820
          GO TO 780                                                     APP03830
750     IF( XR(MI) .GT. QH(6) ) GO TO 760                               APP03840
          DO 755 MKM=6,7                                                APP03850
            K3(MKM) = K3(MKM)+1                                         APP03860
755       CONTINUE                                                      APP03870
          GO TO 780                                                     APP03880
760     IF( XR(MI) .GT. QH(7) ) GO TO 780                               APP03890
          K3(7) = K3(7)+1                                               APP03900
780     CONTINUE                                                        APP03910
800   CONTINUE                                                          APP03920
      DO 410 KK=1,7                                                     APP03930
        PK1(KK) = FLOAT(K1(KK))/FLOAT(KT)                               APP03940
        PK3(KK) = FLOAT(K3(KK))/FLOAT(KT)                               APP03950
410   CONTINUE                                                          APP03960
      DO 420 LM=1,7                                                     APP03970
        PR1(LM) = PR1(LM)+PK1(LM)*NU1((I+1),J)                          APP03980
        PH1(LM) = PH1(LM)+PK3(LM)*NU1((I+1),J)                          APP03990
420   CONTINUE                                                          APP04000
240  CONTINUE                                                           APP04010
250 CONTINUE                                                            APP04020
    DO 425 LL=1,7                                                       APP04030
     IF(PR1(LL) .GT. 1.0) GO TO 200                                     APP04040
      GO TO 201                                                         APP04050
200  PR1(LL) = 1.0                                                      APP04060
201  IF(PH1(LL) .GT. 1.0) PH1(LL) = 1.0                                 APP04070
     DO 424 II=1,M                                                      APP04080
      DO 423 JJ=1,N1                                                    APP04090
       DMY(LL) = PR1(LL)*(1-PR1(LL))*NU1((II+1),JJ)**2/FLOAT(KT)        APP04100
       VARP(LL) = VARP(LL)+DMY(LL)                                      APP04110
       DNY(LL) = PH1(LL)*(1-PH1(LL))*NU1((II+1),JJ)**2/FLOAT(KT)        APP04120
       VPH(LL) = VPH(LL)+DNY(LL)                                        APP04130
423   CONTINUE                                                          APP04140
424  CONTINUE                                                           APP04150
425 CONTINUE                                                            APP04160
490 CONTINUE                                                            APP04170
    DO 310 JJJ=1,N1                                                     APP04180
      X1(1,JJJ) = 0.                                                    APP04190
      A1(1,JJJ) = 0.                                                    APP04200
      B1(1,JJJ) = 0.                                                    APP04210
      C1(1,JJJ) = 0.                                                    APP04220
310 CONTINUE                                                            APP04230
    DO 320 IK=1,M                                                       APP04240
      DO 319 JK=1,N1                                                    APP04250
        X1((IK+1),JK) = R1(IK,JK)                                       APP04260
```

```
          A1((IK+1),JK) = AR1(IK,JK)                                    APP04270
          B1((IK+1),JK) = BR1(IK,JK)                                    APP04280
          C1((IK+1),JK) = CR1(IK,JK)                                    APP04290
 319    CONTINUE                                                        APP04300
 320  CONTINUE                                                          APP04310
      RETURN                                                            APP04320
      END                                                               APP04330
C                                                                       APP04340
C***********************************************************************APP04350
C SUBROUTINE TO READ CLT. MEAN AND VARIANCE                             APP04360
C***********************************************************************APP04370
      SUBROUTINE APPROX(XBLT,SDLT)                                      APP04380
      REAL XBLT,SDLT                                                    APP04390
C READ RESULTS FROM APL PROGRAM GIVEN M,N,LAM1,LAM2,MU1,MU2             APP04400
      XBLT = 7.170168127                                                APP04410
      SDLT = 0.05611889739                                              APP04420
      RETURN                                                            APP04430
      END                                                               APP04440
C                                                                       APP04450
C***********************************************************************APP04460
C SUBROUTINE TO COMPUTE HEAVY TRAFFIC MEAN AND VARIANCE                 APP04470
C***********************************************************************APP04480
      SUBROUTINE HTA2(T,M,N,LAM1,LAM2,MU1,MU2,HTXB,HTSD)                APP04490
      REAL T,LAM1,LAM2,MU1,MU2,HTXB,HTSD,XL1,XL2,XC,AA,BB,CC            APP04500
     *,XM1,XM2,AS1,AS2,BS1,BS2,SM1,SM2,S0,S1,B11,B12,B21,B22            APP04510
     *,K11,K12,K21,K22,G31,G32,B31,B32,K31,K32,Y10,Y20,Y12              APP04520
     *,ZT1,ZT2,DZ1,DZ2,DZ3,DT1,DT2,DT3,DS1,DS2,DS3,D10,D20              APP04530
      INTEGER M,N                                                       APP04540
      XL1 = LAM1*M                                                      APP04550
      XL2 = LAM2*N                                                      APP04560
      XC  = FLOAT(N)/FLOAT(M)                                           APP04570
      AA  = XL1*(MU1*XL2-MU2*XL1)                                       APP04580
      BB  = MU2*XL1**2+(MU1-XL1)*(MU1*XL2-MU2*XL1)+XC*XL2*XL1*MU1       APP04590
      CC  = XL1*MU2*(MU1-XL1)-XL2*XL1*XC*MU1                            APP04600
      IF( MU1*XL2-MU2*XL1 ) 100,110,100                                 APP04610
 100    XM1 = (SQRT(BB**2-4*AA*CC)-BB)/(2*AA)                           APP04620
        GO TO 200                                                       APP04630
 110  XM1 = -CC/BB                                                      APP04640
 200  XM2 = XM1*MU1/(XL1*(1-XM1))-XM1                                   APP04650
C COMPUTE THE MEAN                                                      APP04660
      HTXB = M*(XM1+XM2)*T                                              APP04670
C                                                                       APP04680
      AS2 = -XL1*(1-XM1)                                                APP04690
      BS1 = -XL2*(XC-XM2)                                               APP04700
      AS1 = XL1*(XM1+XM2)+MU1+AS2                                       APP04710
      BS2 = XL2*(XM1+XM2)+MU2+BS1                                       APP04720
      SM1 = SQRT(XM1*MU1-AS2*(XM1+XM2))                                 APP04730
      SM2 = SQRT(XM2*MU2-BS1*(XM1+XM2))                                 APP04740
```

```
      S0  = (SQRT((BS2+AS1)**2-4*(AS1*BS2-AS2*BS1))-BS2-AS1)/2.          APP04750
      S1  = -(BS2+AS1+S0)                                                 APP04760
      B11 =  (S0+BS2)/(S0-S1)                                             APP04770
      B21 =  BS1/(S1-S0)                                                  APP04780
      B12 =  AS2/(S1-S0)                                                  APP04790
      B22 =  (S0+AS1)/(S0-S1)                                             APP04800
      K11 =  (S1+BS2)/(S1-S0)                                             APP04810
      K22 =  (S1+AS1)/(S1-S0)                                             APP04820
      K12 =  AS2/(S0-S1)                                                  APP04830
      K21 =  BS1/(S0-S1)                                                  APP04840
      G31 =  (BS2-BS1)/(S0*S1)                                            APP04850
      G32 =  (AS1-AS2)/(S0*S1)                                            APP04860
      B31 =  (G31*S1+1)/(S0-S1)                                           APP04870
      B32 =  (G32*S1+1)/(S0-S1)                                           APP04880
      K31 =  (G31*S0+1)/(S1-S0)                                           APP04890
      K32 =  (G32*S0+1)/(S1-S0)                                           APP04900
      Y10 =  -SM1**2*(B11**2/(2*S0)+2*B11*K11/(S0+S1)+K11**2/(2*S1))      APP04910
      Y10 =  Y10-SM2**2*(B12**2/(2*S0)+2*B12*K12/(S0+S1)+K12**2/(2*S1))   APP04920
      Y20 =  -SM1**2*(B21**2/(2*S0)+2*B21*K21/(S0+S1)+K21**2/(2*S1))      APP04930
      Y20 =  Y20-SM2**2*(B22**2/(2*S0)+2*B22*K22/(S0+S1)+K22**2/(2*S1))   APP04940
      D10 =  B11*B21/(2*S0)+(K11*B21+B11*K21)/(S0+S1)+K11*K21/(2*S1)      APP04950
      D20 =  B12*B22/(2*S0)+(K12*B22+B12*K22)/(S0+S1)+K12*K22/(2*S1)      APP04960
      Y12 =  -SM1**2*D10-SM2**2*D20                                       APP04970
      DS1 =  B31**2*(EXP(2*S0*T)-1)/(2*S0)+K31**2*(EXP(2*S1*T)-1)/(2*S1)  APP04980
      DS2 =  2*G31*B31*(EXP(S0*T)-1)/S0+2*G31*K31*(EXP(S1*T)-1)/S1        APP04990
      DS3 =  G31**2*T+2*B31*K31*(EXP((S0+S1)*T)-1)/(S0+S1)                APP05000
      ZT1 =  DS1+DS2+DS3                                                  APP05010
      DT1 =  B32**2*(EXP(2*S0*T)-1)/(2*S0)+K32**2*(EXP(2*S1*T)-1)/(2*S1)  APP05020
      DT2 =  2*G32*B32*(EXP(S0*T)-1)/S0+2*G32*K32*(EXP(S1*T)-1)/S1        APP05030
      DT3 =  G32**2*T+2*B32*K32*(EXP((S0+S1)*T)-1)/(S0+S1)                APP05040
      ZT2 =  DT1+DT2+DT3                                                  APP05050
      DZ1 =  (G31+B31*EXP(S0*T)+K31*EXP(S1*T))**2*Y10                     APP05060
      DZ2 =  (G32+B32*EXP(S0*T)+K32*EXP(S1*T))**2*Y20                     APP05070
      DZ3 =  2*SQRT(DZ1/Y10)*SQRT(DZ2/Y20)*Y12                            APP05080
C COMPUTE THE VARIANCE                                                    APP05090
      HTSD =  (DZ1+DZ2+DZ3+ZT1*SM1**2+ZT2*SM2**2)*M                       APP05100
C                                                                         APP05110
      RETURN                                                              APP05120
      END                                                                 APP05130
```

LIST OF REFERENCES

1. Coffman, E.G., Muntz, R.R. and Trotter, H., "Waiting time distribution for processor-sharing system," *Journal of the Association for Computing Machinary*, 17, pp. 120-130, 1970.

2. Mitra, D., *Waiting time distributions from closed queueing network models of shared processor systems*, Bell Laboratories Report, 1981.

3. Gaver, D., Jacobs, P. and Latouche, G., *The Normal Approximation and Queue Control for Response Time in a Processor-Shared Computer System Model*, Naval Postgraduate School Technical Report, 1984.

4. Lavenberg, S.S. and Reiser, M., "Stationary state probabilities at arrival instants for closed queueing network with multiple types of customers," *Journal of Applied Probability*, 17, pp. 1048-1061, 1980.

5. Cohen, J. W., "The multiple phase service network with generalized Processor Sharing," *Acta Informatica*, 12, pp. 245-284, 1979.

6. Cinlar, E., *Introduction to Stochastic Processes*, pp. 269-271, Prentice-Hall, Englewood Cliffs, N.J., 1975.

7. Kelly, F. P., *Reversibility and Stochastic Networks*, p. 12, John Wiley and Sons, New York, 1979.

8. Keilson, J., *Markov Chain Models-Rarity and Exponentiality*, Springer-Verlag, New York, 1979.

9. Cox, D. R., *Renewal Theory*, Methuen Monograph, 1962.

10. Iglehart, D. L., "Limiting diffusion approximations for the many server queue and the repairman problem," *Journal of Applied Probability*, 2, pp. 429-441, 1965.

11. Karlin, S. and Taylor, H. M., *A first course in Stochastic Process*, (second edition), Academic Press, New York, 1975.

12. Gaver, D., Jacobs, P., *Processor-shared time-sharing model in heavy traffic*, in preparation.

13. Arnold, L., *Stochastic Differential Equations : Theory and Applications*, John Wiley and Sons, 1974.

14. Berman, D., An analytical approach to Diffusion Approximation in Queueing, Unpublished Doctoral Dissertation, New York University, 1979.

15. Cochran, W. G., Sampling Techniques, (second edition), John Wiley and Sons, 1963.

16. Gaver, D. P., and Lehoczky, J. P., "Gaussian approximation to service problem: a communications system example," Journal of Applied Probability, 13, pp. 768-780, 1976.

17. Gaver, D., Jacobs, P. and Latouche, G., "Finite birth and death models in randomly changing environments," to appear Journal of Applied Probability, 1984.

18. Kleinrock, L., Queueing System, Vol.II, Wiley-Interscience, 1976.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943	2
3.	Department Chairman, Code 55 Department of Operation Analysis Naval Postgraduate School Monterey, California 93943	1
4.	Professor P. A. Jacobs, code 55Jc Department of Operation Analysis Naval Postgraduate School Monterey, California 93943	1
5.	Professor D. P. Gaver, code 55Gv Department of Operation Analysis Naval Postgraduate School Monterey, California 93943	1
6.	Department of Educations The Headquarters of Royal Thai Navy Tanon Arun Amarin, Bangkok Yai Bangkok 10600, Thailand	2
7.	Ens. Suriya Pornsuriya 155 Moo 4, Tumbon Bangyapraek Amphoe Muang, Samut Sakhon 74000 Thailand	2

207404

Thesis
P7416 Pornsuriya
c.1 Normal approximation
for response time in a
processor-shared computer system model.